MUESAEI

Màster Universitari en Enginyeria de Sistemes Automàtics i Electrònica Industrial

Alumne: Remi Martínez Lara

ESTUDI I IMPLEMENTACIÓ DEL CONTROL D'UNA ESTACIÓ AUTOMATITZADA MITJANÇANT EL RECONEIXEMENT DE VEU

TREBALL DE FI DE MÀSTER

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

Directora del TFM: Rita Maria Planas Dangla

Convocatòria de lliurament: 30 de Juny de 2020

RESUM

El propòsit del present projecte és el desenvolupament d'una aplicació que permeti el control del moviment d'una estació automatitzada mitjançant el reconeixement de veu. El projecte es divideix en tres sectors diferents: el primer, és el desenvolupament d'una aplicació que permeti reconèixer la veu i generar comandes que el PLC pugui interpretar per poder realitzar-les. El segon sector és l'establiment de la comunicació de l'aplicació desenvolupada amb el PLC, per tal que pugui rebre tota la informació necessària per executar les instruccions demanades, i el tercer sector és el desenvolupament de la programació del PLC per interpretar la informació rebuda i poder-la executar de manera satisfactòria.
El primer sector tracta sobre el desenvolupament de l'aplicació de reconeixement de veu amb el llenguatge de programació Python. Es detallen punts com la història del reconeixement de veu, l'anàlisi del seu funcionament i les tècniques utilitzades actualment per aquesta tecnologia. També s'explica que és una API (Interfície de programació d'aplicacions), i la seva funcionalitat pel desenvolupament d'aplicacions de reconeixement de veu. S'explica també la configuració necessària per la correcta funcionalitat del reconeixement de veu, i el desenvolupament de l'aplicació realitzada, la qual genera una senyal d'àudio que és capaç de reconèixer i transcriure aquesta senyal en text.

El segon sector explica els processos necessaris per poder establir una correcta comunicació entre l'aplicació de reconeixement de veu i el PLC. Per poder assolir aquest objectiu, es fa a través d'un servidor OPC amb el software KEPServerEX.

El tercer sector tracta sobre el desenvolupament de la programació del PLC en llenguatge Ladder. S'ha comprovat el correcte funcionament del programa a través d'una simulació aportada pel propi software Sysmac Studio a través d'un sistema HMI desenvolupat.

Degut a la pandèmia ocasionada pel SARS-CoV-2 (COVID-19), el sector dos s'ha vist afectat perquè no s'ha pogut establir comunicació entre el servidor OPC i el PLC, si no que s'ha hagut d'establir comunicació entre el servidor OPC i la simulació creada amb el software Microsoft Excel. Es detalla la configuració necessària per a la connexió entre el servidor OPC i Microsoft Excel, així com el seu funcionament. El sector tres també s'ha vist afectat perquè no s'ha pogut accedir al laboratori de Robòtica i CIM i, per tant, no ha sigut possible comprovar el funcionament en un entorn real. És per això que s'ha creat una simulació de l'estació automatitzada amb el software Microsoft Excel amb Visual Basic.

Finalment, es detalla la guia d'operació i de connexió de cadascun dels programes generats en aquest projecte.

PARAULES CLAU

PLC, Python, Speech Recognition, Servidor OPC, Reconeixement de veu, PyAudio, Ladder, API Google, Servidor OPC UA, Servidor OPC DDE

AGRAÏMENTS

Primer de tot, m'agradaria agrair al Màster Universitari en Enginyeria de Sistemes Automàtics i Electrònica Industrial i a la tutora d'aquest projecte Rita Maria Planas Dangla, ja que, han pogut donar-me els coneixements necessaris per poder realitzar aquest projecte.

Acabar aquest Màster és un punt d'inflexió molt important ja que, gràcies als valors adquirits amb el pas dels anys i del temps, em sento preparat per afrontar una vida plena d'objectius. Per això mateix, m'agradaria agrair als companys que he anat trobant al llarg de l'etapa universitària, als meus amics, que m'han ajudat quan ho he necessitat i, sobretot, a la meva família, que ha estat sempre al meu costat, ajudant-me i motivant-me per continuar endavant.

A la meva mare, qui ha fet tot el que ha estat a les seves mans per ajudar-me i de ben segur serà immensament feliç amb aquesta fita. Al meu pare, qui ha sigut el meu referent i inspiració per saber que volia estudiar una enginyeria des de ben petit. Sempre que he necessitat alguna cosa, el meu pare ha estat allà per ajudar-me. Al meu germà, qui ha patit amb mi llargues nits d'estudi i m'ha donat la motivació per continuar estudiant i formant-me, perquè també ha sigut un referent meu des de petit. A la meva novia, qui també ha estat sempre al meu costat, recolzant-me incondicionalment en tot moment. Gràcies a tots, he arribat fins a on estic avui dia i estic segur que sense tot el que m'ha aportat cadascun d'ells i d'elles, no ho hauria aconseguit. Per això i molt més, els meus agraïments més profunds per a tots i totes.

El meu treball de final de Màster ho dedico amb moltes ganes a totes les persones que han estat amb mi incondicionalment i a qui m'ha recolzat i motivat per continuar endavant. Als meus pares, el meu germà, la meva novia, els meus amics i als professors i professores que han format part de la meva vida en aquesta etapa.

INDEX

INDEX D'IL·LUSTRACIONS

INDEX DE TAULES

1. INTRODUCCIÓ

Aquesta part mostra un anàlisi general del projecte desenvolupat, en la qual es detallen els punts inicials, la proposta de treball, i una descripció de com el projecte ha estat estructurat pel seu desenvolupament.

1.1 JUSTIFICACIÓ

Aquest projecte consta de dos àrees principals les quals avui dia estan en constant creixement i són peces clau pel desenvolupament tecnològic i pel desenvolupament de sistemes automàtics aplicats a indústries i processos productius. Aquestes àrees són l'automatització industrial i les comunicacions industrials.

L'automatització comprèn el conjunt de recursos que impliquen equips físics (hardware) i programes destinats al control d'aquests equips (software), mètodes i tecnologia encaminats al control efectiu d'un determinat procés, de manera que aquest es dugui a terme d'una manera automàtica, reduint en tot el possible la intervenció humana. El propòsit de l'automatització és dur a terme processos industrials automàtics, en un entorn de producció eficient, amb tendència a satisfer la creixent demanda de béns, en intervals de temps cada vegada més reduïts (Vicuña, Arrizibita i Unzué, 2019).

Amb l'entorn en el que es viu avui dia econòmicament globalitzat i fort competitivament, l'automatització s'ha convertit en un factor clau a l'hora d'optimitzar els processos productius i poder satisfer la demanda cada cop a menys preu i amb temps d'entrega més reduïts.

Les comunicacions per a la transferència de dades entre tots els elements de l'ecosistema productiu industrial són la base del nou paradigma Indústria 4.0. Es comuniquen els controladors o PLCs entre sí, amb entrades o sortides digitals o analògiques distribuïdes, amb el sistema de supervisió. També, aigües avall, amb instruments i sensors i, aigües amunt, amb sistemes de gestió, amb bases de dades massives i amb el núvol. El mateix producte que es fabrica és part de l'ecosistema i també es comunica amb ell, traçant des de l'inici tota la genètica de la seva vida útil. I per últim però no per això menys important, l'home, que s'incorpora com a eina intel·ligent, escassa però necessària en els processos de producció. És per aquesta enorme demanda de dades entre elements que són necessàries diferents formes i interfícies de comunicació. De vegades, es necessiten enllaços amb enormes amples de banda que permetin comunicar en temps molt curts, gran quantitat de dades. D'altres, necessitem enllaços "sense cables" per a elements mòbils o allunyats.

I n'hi ha, a les preval el baix consum de l'element comunicador (Automática e Instrumentación, 2020).

A l'actualitat existeixen moltes aplicacions en diferents àmbits en els quals s'ha fet una integració de sistemes de reconeixement de veu on ha contribuït a tenir una millora en l'experiència de l'usuari ja que són aplicacions intuïtives i fàcils d'utilitzar. És un tema que està en constant desenvolupament, i avui dia és un factor clau a nivell de desenvolupament i creixement empresarial. És per això, que s'ha decidit realitzar aquest projecte, en el qual es desitja integrar el sistema de reconeixement de veu al control del moviment d'una estació automatitzada, d'aquesta manera tindrà un ús més accessible per a usuaris que no siguin experts en el tema de l'automatització.

1.2 OBJECTIUS

El treball té com a objectiu principal l'estudi de la viabilitat i la posterior implementació del control d'una estació automatitzada utilitzant el reconeixement de veu. Caldrà utilitzar un software de reconeixement de veu, per poder interpretar i reconèixer les ordres donades verbalment i comunicar-les a un PLC que és l'encarregat de dur a terme el control d'una estació de procés.

Els objectius específics del projecte són els següents:

- Estudi, control i programació d'una Estació Automatitzada – PLC Omron NJ mitjançant el software Sysmac Studio.

- Estudiar la tecnologia de reconeixement de veu i desenvolupar una aplicació a través de Python que sigui capaç de reconèixer comandes de veu, i donar com a resposta una sèrie d'instruccions que puguin ser interpretades pel PLC, per a realitzar una acció determinada.

- Establir comunicació entre l'aplicació del reconeixement de veu i el PLC a través d'un servidor OPC.

- Programació mitjançant Microsoft Excel (vba) de la simulació de l'Estació Automatitzada per a la posterior comunicació entre l'aplicació de reconeixement de veu i el Software Microsoft Excel a través d'un servidor OPC.

1.3 ABAST

El present projecte de final de màster, comprèn tots els punts involucrats amb l'anàlisi del funcionament del sistema de reconeixement de veu, la programació de l'aplicació, la programació de la simulació de l'estació automatitzada, la interconnexió de tots els elements a través d'un servidor OPC, tant en entorn real com en la simulació i la posada en marxa del sistema de l'estació automatitzada del laboratori de Robòtica i CIM.

El desenvolupament del sistema de reconeixement de veu serà executat a través del software Python, ja que és un dels llenguatges de programació més utilitzats en l'actualitat, té un llenguatge versàtil, multi-plataforma i comporta una sintaxis que afavoreix que el codi sigui llegible. L'aplicació serà executada en un mini-ordinador, anomenat placa LattePanda, la qual té un sistema operatiu Windows 10, i la senyal d'àudio serà adquirida mitjançant un micròfon extern.

La programació de l'entorn real de l'estació automatitzada, és a dir, del PLC Omron-NJ, serà en llenguatge de programació Ladder. Software en el qual es poden realitzar simulacions, ja sigui amb el propi Ladder o també amb una pantalla HMI del sistema realitzat, de manera que es pot veure el correcte funcionament del sistema. El codi desenvolupat tindrà el propòsit de realitzar la seqüència que requereix l'estació II del laboratori de Robòtica i CIM, ja que és on es desenvoluparà aquest treball de final de màster.

La programació de la simulació de l'estació automatitzada es realitzarà a través del software Microsoft Excel, amb el llenguatge de programació Visual Basic. El propòsit d'aquest codi serà el mateix que el realitzat amb el llenguatge Ladder, és a dir, realitzar la seqüència que requereix l'estació II del laboratori de Robòtica i CIM.

La comunicació entre l'aplicació del reconeixement de veu i les variables del programa desenvolupat en Ladder o Visual Basic, serà establerta a través d'un servidor OPC, KEPServerEX i un client OPC creat per poder-ho executar amb Python.

2. DESCRIPCIÓ GENERAL DEL PROJECTE

2.1 DESCRIPCIÓ DEL PROJECTE

En la descripció general del projecte es descriuen totes les eines que s'han utilitzat per dur a terme tota l'elaboració del projecte, tant hardware com software. Primerament, s'analitzen totes les eines físiques i en la següent secció, s'analitzen tots els programes utilitzats per al desenvolupament del projecte.

2.2 HARDWARE UTILITZAT

A continuació es detallen les eines utilitzades en aquest projecte, així com una breu descripció de cadascuna i com són utilitzades. Primerament, tot el projecte està integrat en una placa LattePanda que té el sistema operatiu de Windows 10. L'obtenció de la senyal d'àudio es fa a través d'un micròfon, i la comunicació amb el PLC Omron NJ es fa a través del servidor KEPServerEX.

Pel fet de no haver pogut accedir a l'estació II del laboratori de Robòtica i CIM, s'ha creat un simulador amb Microsoft Excel i visual basic, i en aquest cas, la comunicació de la senyal d'àudio serà creada entre el simulador amb Microsoft Excel i el servidor KEPServerEX.

2.2.1 MICRÒFON

Una de les parts principals del projecte és el micròfon, ja que per al funcionament correcte de l'estació, és necessari obtenir la senyal d'entrada i poder enregistrar les instruccions de veu al sistema.

És molt important que el micròfon tingui la major qualitat de so possible, d'aquesta manera es redueix el soroll que pot produir-se al entorn de treball o a la targeta de processament de la senyal d'entrada.

En aquest cas, el micròfon utilitzat serà el d'uns auriculars Mad Catz f.r.e.q. TE. Són uns auriculars circumaurals, és a dir, cobreixen la orella de l'operari o l'usuari que farà ús d'aquest micròfon, i els quals aïllen completament el so de l'exterior.

En la il·lustració 1 es mostra el micròfon que s'utilitzarà en aquest projecte:

Il·lustració 1. Auriculars amb micròfon Mad Catz f.r.e.q. TE

2.2.1.1CARACTERÍSTIQUES

Pel que respecta a les característiques del micròfon, cal destacar:

- L'escuma còmoda de memòria protegeix les orelles
- El micròfon pot ser silenciat, el volum es pot canviar o una trucada entrant es pot contestar
- El micròfon té una alta qualitat de so
- Té sistema de micròfon dual per a una comunicació precisa
- Compatible amb la majoria d'ordinadors amb ports USB o Jack, ja que, aquests auriculars venen amb un adaptador que converteix el connector Jack en USB

2.2.1.2 ESPECIFICACIONS

A continuació es mostraran les especificacions del micròfon que s'utilitzaran en aquest projecte pel reconeixement de veu:

- Audífons: circumaural
- Cables inclosos: Mini-USB
- Canals de sortida d'àudio: 7.1 canals
- Color del producte: Negre
- Connector USB: Mini-USB
- Connector de 3,5 mm: Si
- Connexió USB: Si
- Connexió a PC: USB 2.0

- Freqüència d'auriculars: 20 - 20000 Hz
- Freqüència de micròfon: 200 - 5000 Hz
- Longitud de cable: 1 m
- Longitud del cable allargador: 2,5 m
- Sistema operatiu Windows suportat: Windows 7, Windows 8 i Windows 10
- Tecnologia de connectivitat: amb fil
- Tipus d'auricular: binaural

2.2.2 PLACA LATTEPANDA

El projecte està implementat en una placa LattePanda, la qual pot executar una versió completa de Windows 10. La placa compta amb un processador Intel Quad Core i té una connectivitat excel·lent, amb tres ports USB, Wi-Fi i Bluetooth 4.0 integrats. També inclou un coprocessador Arduino que permet interactuar amb el món físic al disposar d'entrades i sortides digitals o analògiques. A continuació es mostra una imatge de la placa LattePanda (LattePanda, 2019):

Il·lustració 2. Placa Lattepanda (LattePanda, 2019)

2.2.2.1 CARACTERÍSTIQUES

Les principals característiques de la placa Lattepanda són les següents (LattePanda, 2019):

- Versió completa del Sistema operatiu Windows 10

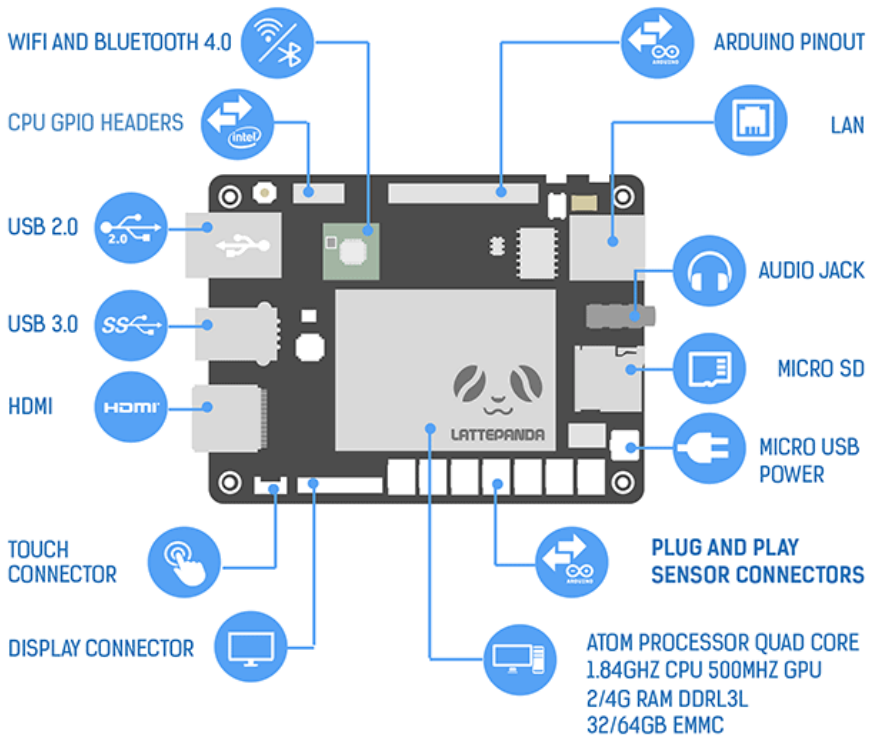
- LattePanda és diferent del Raspberry Pi i d'altres taulers de desenvolupament ja que suporta un sistema complet de Windows 10. Amb recursos de programari abundants i un ecosistema madur de Windows a la vostra disposició, LattePanda proporciona a les vostres idees més accessibilitat i potència.

- És una placa versàtil, petita, potent i barata amb l'objectiu d'obrir una nova era informàtica per als usuaris ja que pot servir tant per a ús personal com per projectes de robòtica, electrònica, etc.

- LattePanda aporta ordinadors de placa única a un nivell de potència i rendiment completament nou. Processador turbo-alimentat Intel Quad Core 1.8GHz, 2-4 GB de memòria RAM i 32-64 GB de memòria flash a bord, LattePanda pot realitzar fàcilment el reconeixement d'imatges, control CNC en temps real i molt més

- LattePanda no és només un ordinador regular de Windows de baix cost, sinó que també inclou un coprocessador Arduino, cosa que significa que es pot utilitzar per controlar i intuir el món físic quan s'afegeixen sensors i actuadors.

A continuació es mostra una imatge en la qual es veu al detall cadascun dels components de la placa LattePanda:



Il·lustració 3. Característiques de la placa LattePanda (LattePanda, 2019)

2.2.2.2 ESPECIFICACIONS

A continuació es detallen les especificacions del hardware de la placa LattePanda:

- Processador: Intel Cherry Trail Z8350 Quad Core, memòria cau 2M, fins a 1,92 GHz
- Sistema operatiu: Windows 10 preinstal·lat
- RAM: 4GB DDR3L
- Capacitat d'emmagatzematge: 64 GB
- GPU: Intel HD Graphics, 12 EUs @ 200-500 Mhz, memòria d'un sol canal
- Un port USB3.0 i dos ports USB 2.0
- WiFi i Bluetooth 4.0
- Co-processador Arduino integrat: ATmega32u4
- Sortida de vídeo: HDMI i MIPI-DSI
- Connector de superposició del panell tàctil
- Compta amb 100Mbps Ethernet
- GPIO:
    - 6 GPIO del processador Cherry Trail
    - 20 GPIOs d'Arduino Leonardo
    - 6 Connectors i reproductors del sensor de gravetat

- Potència: 5v / 2A
- Dimensió del tauler: 88 * 70 mm / 3.46 * 2.76 polzades
- Mida de l'envasament: 110 * 94 * 30 mm / 4.33 * 3.70 * 1.18 polzades
- Pes: 55g

2.2.3 PLC OMRON-NJ

PLC (Programmable Logic Controller), és a dir, controlador lògic programable és el dispositiu que permet l'automatització d'un procés electromecànic. Es tracta d'un ordinador que s'encarrega de controlar el funcionament de les màquines emprades en la producció o en el muntatge (Pérez, 2019).

Un PLC pot treballar amb amplis rangs de temperatura, és resistent als impactes i a les vibracions i pot gestionar una gran quantitat de senyals de sortida i d'entrada de manera simultània (Pérez, 2019).

En l'actualitat, un PLC disposa de capacitats similars a les que tenen els ordinadors. Poden controlar diversos processos i moviments i les comunicacions per xarxa, per exemple (Pérez, 2019).

El component essencial del PLC és la unitat central de processament (CPU). Es troba integrada pel processador i per la memòria que executa el programa. El PLC també disposa de les interfícies d'entrada (que reben senyals procedents de claus, botons, etc.) i de sortida (per controlar vàlvules, motors i altres actuadors) (Pérez, 2019).

En definitiva, és un ordinador especial que s'utilitza a nivell industrial per al processament de les dades de les màquines. Gràcies a aquests elements, es poden controlar automàticament tot tipus de processos industrials (Pérez, 2019).

El PLC Omron NJ, és la base principal d'aquest projecte, ja que sense el PLC no es podria dur a terme la realització del mateix.

Il·lustració 4. PLC Omron NJ (Omron)

2.2.3.1 CARACTERÍSTIQUES

Les característiques principals del PLC Omron NJ són les següents (Omron):

- Temps de cicle més ràpid: 1 ms.
- Nombre d'eixos: 2.
- Nucli de motion sincronitzat.
- Funcions: lògica, motion i connexió amb base de dades.
- La connexió directa a la base de dades s'amplia en la versió 2.0 amb la possibilitat de realitzar trucades als procediments emmagatzemats i funcionalitat de comunicacions segures.
- Connexió directa amb base de dades: client SQL per servidor Microsoft SQL, Oracle, IBM DB2, MySQL, Firebird, PostgreSQL.
- Multitasca.
- Ports EtherCAT i Ethernet / IP integrats.
- Connexió amb base de dades: client SQL per a Microsoft SQL Server, Oracle, IBM DB2, MySQL, Firebird.
- La topologia d'anell EtherCAT és compatible per mantenir les comunicacions i el control en cas que es trenqui un cable o falli un dispositiu.

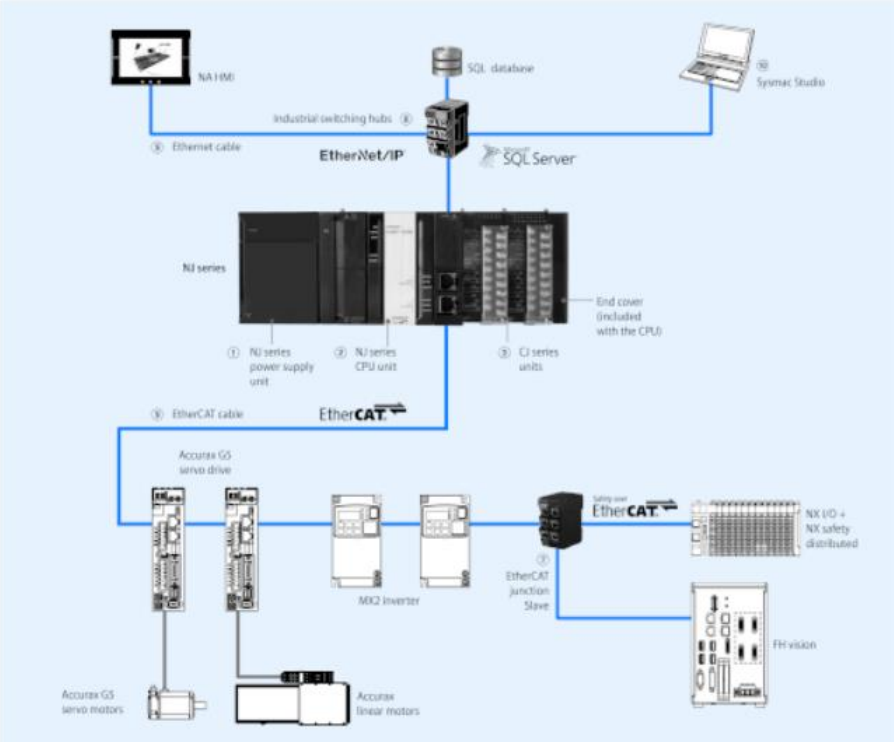
2.2.3.2 ESPECIFICACIONS

A continuació es mostraran les especificacions obtingudes de la documentació del PLC Omron NJ (Omron):

- Rack de CPU: 10 unitats màx.
- Rack d'expansió: 10 unitats màxim, fins a tres racks d'expansió
- 40 unitats màx. per sistema (rack de CPU + 3 rack d'expansió)
- Consum actual: 1,90 A a 5 VCC

- Unitat d'alimentació de 100 a 240 VAC per a CPU NJ

La il·lustració 5 mostra el diagrama de connexions del PLC NJ:



Il·lustració 5. Esquema de connexió de PLC Omron NJ (Omron)

2.2.4 ESTACIÓ AUTOMATITZADA

L'estació automatitzada d'aquest projecte és l'estació II del laboratori de Robòtica i CIM del TR11. És l'encarregada de foradar les peces subministrades per l'estació I, per això disposa d'un cilindre de doble tija que és l'encarregat de recollir i expulsar les peces de l'estació de manera simultània. Aquestes peces es depositen sobre un plat giratori que les posiciona a sota d'un taladro que serà l'encarregat de foradar les peces, les quals han sigut prèviament mitjançant un cilindre de simple efecte.

L'estació està formada pels següents elements:

CINTES

- Cinta d'entrada de Peces
- Cinta de sortida de Peces

MOTORS

- Motor del Plat Giratori M1
- Motor de Taladro M2
- Motor cinta M3

CILINDRES

- Cilindre A de doble tija
- Cilindre B de fixació del Plat Giratori
- Cilindre C de baixada del Taladro
- Ventosa D

SENSORS

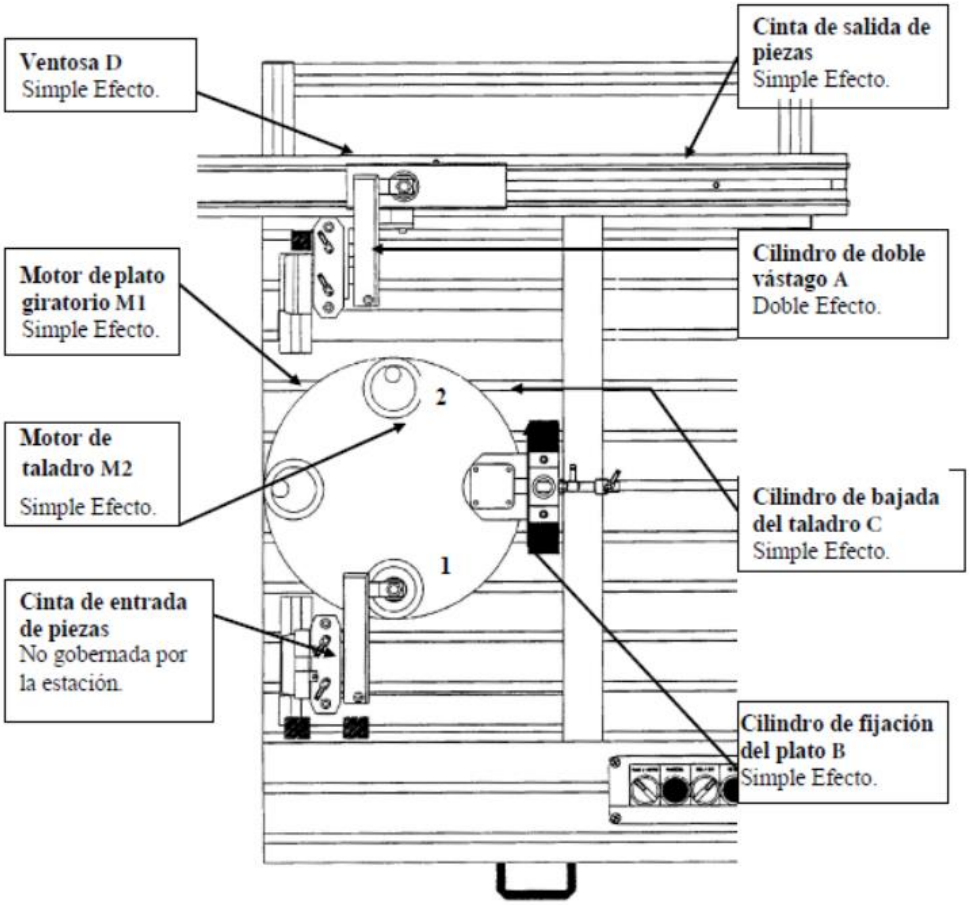
- Sensor posició del plat giratori
- Sensor de presència de peça en la cinta
- Sensor de presència de peça en el plat
- Vàlvula progressiva d'aire comprimit
- Sensor de Buit

BOTONS

- Selector Automàtic/Manual
- Botó Marcha
- Botó Reset
- Pulsador Parada d'Emergència

Tant els cilindres B, C i D, com els motors M1, M2, M3 i la vàlvula d'aire comprimit, tots són de simple efecte, és a dir, es desplacen en un únic sentit, ja que fan la feina en una sola carrera de cicle. En canvi, només el cilindre A de doble tija és de doble efecte, és a dir, és un cilindre capaç de produir treball útil en els dos sentits, ja que disposa d'una força activa tant en l'avanç com en el retrocés.

En la il·lustració 6 es pot veure un esquema orientatiu de l'estació automatitzada:



Il·lustració 6. Esquema Estació II Laboratori Robòtica i CIM

2.3 SOFTWARE UTILITZAT

2.3.1 PYTHON

El llenguatge de programació utilitzat en aquest projecte és Python. És un llenguatge de programació versàtil, multi-plataforma i multi-paradigma, el qual ha estat utilitzat en moltíssimes aplicacions i sistemes operatius. A més, és un llenguatge d'iniciació de molts programadors ja que és impartit en àmbit acadèmic.

Python és un llenguatge de programació interpretat el qual la seva filosofía comporta una sintaxis que afavoreix que el codi sigui llegible. Defineix aquesta programació com multi-paradigma, degut a que soporta orientació a objectes, programació imperativa i en menor mesura programació funcional (Laca, 2020).

Els llenguatges interpretats són aquells que el codi del programador és traduït mitjançant un intèrpret a mesura que es necessari. El fet que Python sigui interpretat presenta certs avantatges com que no es necessita compilar, això

estalvia temps en el seu desenvolupament i prova de l'aplicació, i el codi font pot ser executat a qualsevol software sempre i quan aquest disposi de l'intèrpret (Windows, Linux, Mac, Android, Web) (Laca, 2020).

A continuació es presenten les característiques principals de Python (Laca, 2020; Robledano,2019):

- Multi-paradigma: Python és un llenguatge que suporta més d'un model de desenvolupament (paradigma).

- Imperatiu: Descriu l'estat del programa i permet la seva modificació mitjançant condicions o instruccions de codi que indiquen a l'ordinador com realitzar una tasca. A la programació imperativa es descriu pas a pas un conjunt d'instruccions que s'han d'executar per canviar l'estat del programa i solucionar el problema.

- Funcional: És un paradigma que es basa en l'ús de funcions matemàtiques que permet la variació del programa mitjançant la mutació de variables. Això permet operar amb dades d'entrada i de sortida, podent ingressar dades que seran processades per poder donar altres dades de sortida.

- Orientat a Objectes (POO): Aquesta programació ofereix la particularitat en la forma d'obtenir resultats. Els objectes manipulen els objectes d'entrada per a la obtenció de resultats (sortida) específics, on cada objecte ofereix una funció específica i també permet l'agrupació de biblioteques o llibreries.

- De tipus Dinàmic: És de tipus dinàmic quan una variable pot prendre diferents valors de diferents tipus en moments diferents. A Python les variables són declarades pel seu contingut i no pel seu contenidor. Això permet canviar el valor i tipus d'una variable durant la seva execució sense necessitat de tornar a declarar.

- Simplificat i ràpid: Aquest llenguatge simplifica molt la programació.

- Elegant i flexible: El llenguatge ofereix moltes facilitats al programador al ser fàcilment llegible i interpretable.

- Programació sana i productiva: Es senzill d'aprendre, amb una corba d'aprenentatge moderada. Es molt fàcil començar a programar i fomenta la productivitat.

- Ordenat i net: És un llenguatge molt ordenat amb els mòduls ben organitzats.

- Portable: Es pot utilitzar pràcticament a qualsevol sistema de l'actualitat, ja sigui MAC OS X, Windows, Linux, ...

- Python té una gran biblioteca estàndard que admet moltes tasques de programació comunes, com connectar-se a servidors web, llegir i modificar arxius, etc.

- Té un entorn en desenvolupament inclòs que es diu IDLE.

- Open Source: aquest llenguatge de programació és de programari lliure, significa que pots emprar-ho en qualsevol moment per als teus projectes. A més, pots utilitzar-ho com a base per a crear extensions o desenvolupar mòduls.

Totes aquestes característiques són les que fan Python el llenguatge de programació ideal per poder desenvolupar el projecte pel que fa al reconeixement de veu.

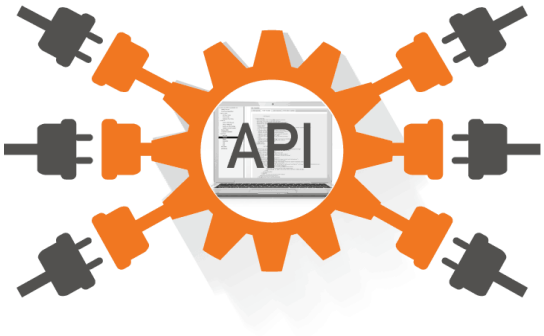
2.3.2 INTERFÍCIE DE PROGRAMACIÓ D'APLICACIONS (API)

Una API (Application Programming Interface), és a dir, una interfície de programació d'aplicacions és un conjunt d'ordres, funcions i protocols informàtics que permet als desenvolupadors crear programes específics per a certs sistemes operatius (ABC, 2015).

Es tracta del conjunt de trucades a certes biblioteques que ofereixen accés a serveis des dels processos i representa un mètode per aconseguir abstracció en la programació, generalment entre els nivells o capes inferiors i superiors del software (Suarez, 2018).

D'aquesta manera, les API simplifiquen en gran mesura el treball d'un programador, ja que no ha d'escriure el codi des de cero. Aquestes permeten al programador utilitzar funcions ja existents per interactuar amb el sistema operatiu o amb un altre programa.

Les APIs s'utilitzen per fer ús de funcions ja existents en un altre software. Al mateix temps, diferents aplicacions poden fer ús de les API de cadascuna per mantenir comunicació de dades entre elles, de manera que l'usuari no ho pugui veure.



Il·lustració 7. Interfície de programació d'aplicacions (API)

A partir de l'any 2000, va aparèixer el terme que avui dia es conegut com API REST (Representational State Transfer) revolucionant la enginyeria del Software. API REST és qualsevol interfície entre sistemes que utilitzi HTTP per obtenir dades o generar operacions sobre aquestes dades en tots els formats possibles com XML o JSON (Barroso, 2016).

En l'actualitat no existeix cap projecte o aplicació que no disposi d'una API REST per a la creació de serveis professionals a partir del Software. Empreses com Twitter, Youtube, Google, Amazon, Facebook... i centenars d'empreses més que generen negoci gràcies a REST i a les APIs REST.

En el cas d'aquest projecte s'utilitza una API de Google. Les APIs de Google són un conjunt d'APIs desenvolupades per Google, les quals permeten la comunicació i integració dels Serveis de Google amb altres serveis. Per exemple s'inclouen les API de Recerca, Gmail, Traductor o Maps. Les aplicacions de tercers poden utilitzar aquestes API per estendre la funcionalitat dels seus serveis.

Les API proveeixen funcionalitats com anàlisi, aprenentatge automàtic (machine learning), o accés a les dades d'usuari (on estiguin establerts els permisos de lectura).

L'API de Google utilitzada en el cas d'aquest projecte és la de Speech to text, la qual presenta les següents característiques (Google Developers):

- Adaptació de veu: Proporciona una llista amb paraules o amb expressions que serveixin com "suggeriments" per reconèixer frases concretes de l'entrada d'àudio. Es poden fer servir classes per convertir automàticament els nombres en direccions, en anys o en monedes, o per fer un altre tipus de conversions en funció del context.

- Reconeixement automàtic de veu: El reconeixement automàtic de veu es basa en xarxes neuronals d'aprenentatge profund i permet que funcionin certes aplicacions, com les de recerca per veu o les de transcripció.

- Vocabulari internacional: Reconeix més de 120 idiomes i variants amb un ampli vocabulari.

- Transcripció d'àudios en temps real o gravats prèviament: El so pot procedir del micròfon d'una aplicació, però també es pot enviar des d'arxius d'àudio gravats prèviament (inserit o mitjançant Google Cloud Storage). S'admeten diverses codificacions d'àudio, com FLAC, AMR, PCMU i LINEAR16.

- Detecció automàtica d'idioma: En les situacions multilingües, es pot especificar entre dos i quatre codis d'idioma. Cloud Speech-to-Text identifica correctament l'idioma que es parla i proporciona una transcripció.

- Tractament del soroll: És capaç de processar àudio de multitud d'entorns sorollosos sense necessitat de reducció addicional del soroll.

- Filtrat de contingut inadequat: Es pot reduir el contingut inadequat en els resultats de text en alguns idiomes.

- Puntuació automàtica: Puntua les transcripcions amb precisió (comes, punts, signes d'interrogació, etc.) gràcies a l'aprenentatge automàtic.

- Selecció de models: Per optimitzar la transcripció segons el cas pràctic, es pot triar un dels quatre models predefinits: ordres de veu i recerca per veu, telefonia, transcripció de vídeo i predeterminat.

- Registre de la informació dels interlocutors: L'API pot identificar de forma automàtica a què interlocutor pertany cada intervenció en una conversa perquè es pugui saber qui va dir què.

- Reconeixement multicanal: En alguns enregistraments intervenen diversos interlocutors, les veus es graven en canals independents; per exemple, trucades telefòniques amb dos canals o videoconferències amb quatre. En aquests casos, Cloud Speech to Text reconeix cada canal per separat i anota les transcripcions perquè segueixin l'ordre real.

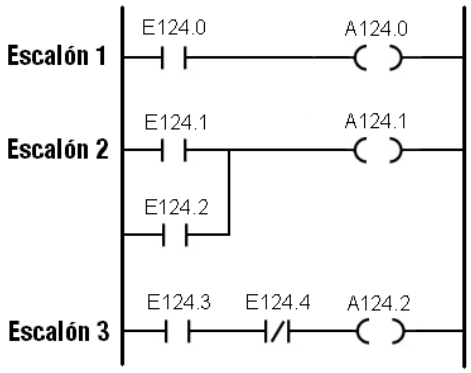
El funcionament de la API Speech to text és bastant senzill. Primerament, s'obté un arxiu d'àudio que s'ha obtingut mitjançant el micròfon, després Python puja l'arxiu d'àudio obtingut a processar al servidor de Google, el qual dóna la transcripció del que s'ha escoltat. La sortida de text es gestionada directament per Python que pot utilitzar aquesta informació per realitzar diferents tasques.

Il·lustració 8. Funcionament API Google Speech to text

2.3.3 LADDER

Ladder és un llenguatge de programació gràfic molt popular dins dels autòmats programables (PLC) ja que està basat en els esquemes elèctrics de control clàssics. D'aquesta manera, amb els coneixements que tot tècnic o enginyer elèctric posseeix, és molt fàcil adaptar-se a la programació en aquest tipus de llenguatge (NochesDeCode, 2011).

Com es pot observar a la il·lustració 9, en aquest tipus de programació, tots els blocs estan situats entre dues línies verticals, una a l'esquerra i l'altra a la dreta. La línia vertical situada a l'esquerra és la que representa un conductor de tensió, i la línia vertical situada a la dreta representa el terra.



Il·lustració 9. Exemple de llenguatge de programació Ladder

A diferència dels diagrames elèctrics, en Ladder totes les accions succeeixen de manera seqüencial, en la manera que s'ha descrit als "esglaons". Els contactes són els elements que activen o desactiven les sortides als "esglaons", els quals provenen d'entrades al PLC. Aquests elements són variables lògiques o binaries, les quals només tenen dos estats: 1 o obert, i 0 o tancat.

Hi ha dos tipus de sortides al programa del PLC, les sortides físiques cap a l'exterior i les que també es coneixen com "relés interns", els quals són variables lògiques que normalment s'utilitzen per memoritzar estats o també es poden utilitzar com acumuladors.

També existeixen funcions lògiques més complexes com poden ser temporitzadors, comptadors, registres de desplaçament, operacions aritmètiques, etc.. les quals són representades en format de blocs.

El que s'ha de tenir en compte d'aquests blocs és el següent (Universitat d'enginyeria UNLP) :

- La base dels temps i el temps final en el cas dels temporitzadors.
- El mòdul de comptatge i condicions de parada i "reset" en el cas dels comptadors.
- Existeixen també blocs funcionals complexes que permeten la manipulació de dades i les operacions amb variables digitals de varis bits.

Per programar un autòmat programable amb Ladder, s'ha de conèixer bé les regles dels circuits de commutació o Lògica de contactes, i a més cal també conèixer els elements que conformen aquest llenguatge. El seu principal avantatge és que els símbols bàsics estan normalitzats segons l'estàndard IEC i són emprats per tots el fabricants.

A continuació es descriuen els elements bàsics més comuns (NochesDeCode, 2011):

- Contacte NA: S'activa quan hi ha un 1 lògic en l'element que representa; és a dir, una entrada (per captar informació del procés a controlar), una variable interna un bit de sistema.
- Contacte NC: La seva funció es similar a la del contacte NA, però en aquest cas s'activa quan hi ha un 0 lògic en l'element que representa.
- Bobina NA: S'activa quan la combinació que hi ha a la seva entrada dóna un 1 lògic. Sol representar elements de sortida, encara que a vegades pot fer el paper de variable interna.

- Bobina NC: S'activa quan la combinació que hi ha a la seva entrada dóna un 0 lògic. El seu comportament es complementari al de la bobina NA.

- Bobina SET: Un cop activa (valor 1) no es pot desactivar (valor 0) si no és per la seva corresponent bobina en RESET. Serveix per memoritzar bits i generalment s'utilitza juntament amb la bobina RESET, les quals juntes donen una enorme potència en la programació.

- Bobina RESET: Permet desactivar una bobina SET prèviament activada.

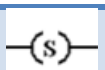
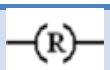
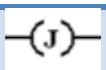
- Bobina JUMP: Permet saltar instruccions del programa i anar directament a l'etiqueta que es desitgi. Serveix per realitzar subprogrames.

Aquests elements s'indiquen mitjançant els caràcters "B" o "M". El seu número d'identificació oscil·la generalment entre 0 i 255. La utilitat fonamental d'aquests elements és la d'emmagatzemar informació per simplificar esquemes i programació.

Els bits de sistema són contactes que el propi autòmat activa quan convé o quan es donen les circumstàncies determinades.

Els contactes poden representar, la entrada d'informació, com per exemple l'activació d'un sensor o l'activació d'una memòria interna.

Les bobines, en canvi, representen aquestes memòries així com les sortides a l'exterior. Activen o desactiven elements com poden ser una llum, un motor elèctric, un solenoide d'una vàlvula pneumàtica, etc.

Contacte NA	Contacte NC	Bobina NA	Bobina NC	Bobina SET	Bobina RESET	Bobina JUMP
						

Taula 1. Elements bàsics de Ladder

- Temporitzadors:

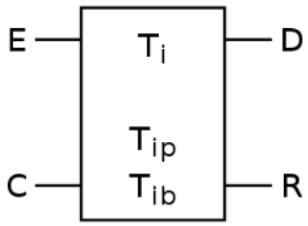
El temporitzador és un element que permet fer activacions amb retard o amb una duració preestablerta amb un control precís del temps d'activació.
L'esquema bàsic pot variar d'un autòmat a un altre, però hi ha una sèrie de senyals que sempre hi són, aquestes són:

- Entrada Enable (E): Activa el procés de recompte de temps. Respon tant a flanc com a nivell lògic. Només es manté comptant mentre E és un "1" lògic, i al detectar flanc de baixada el comptador es restableix.

- Entrada Count (C): Indica el valor de temporització. Ha d'estar activa a "1" lògic en tot moment durant l'interval de temps, ja que si es desactiva ("0" lògic) s'interromp el recompte.

- Sortida Delayed/TON (D): Sortida a "0" per defecte. Es posa a "1" quan el comptador porta activat (enabled) un valor de temps equivalent al valor de temporització o retard (C), i es manté encès mentre la entrada E estigui a "1".

- Sortida TOFF (R): Sortida a "0" per defecte. Es posa a "1" quan es detecta un flanc de baixada a la entrada Enable, i es manté a "1" durant el temps especificat per C.



Il·lustració 10. Esquema d'un temporitzador Tii

A la majoria dels llenguatges Ladder, els comptadors On-delay i Off-delay són entitats independents, per tant, només es tindria una sortida TON o una TOF. El valor del recompte normalment està especificat per programa i no per entrades, per tant el temporitzador es finalment un element amb una única entrada (Enable) i una única sortida (TON/TOF).

- Comptadors:

El comptador és un element capaç de portar el còmput de les activacions de les seves entrades, de manera que resulta adequat per memoritzar successos que no tinguin a veure amb temps però que es necessitin realitzar un determinat nombre de vegades.

A continuació es veuran les diferents senyals per les quals està compost un comptador:

- Entrada RESET (R): Permet posar a 0 el comptador cada vegada que s'activa. Generalment s'utilitza al principi de la execució

assignant-li els bits d'inici, de manera que quedi a 0 cada vegada que s'inicia el sistema.

- Entrada PRESET (P): Permet posar el compte del comptador a un valor determinat diferent de 0, que prèviament s'ha programat en Cip.

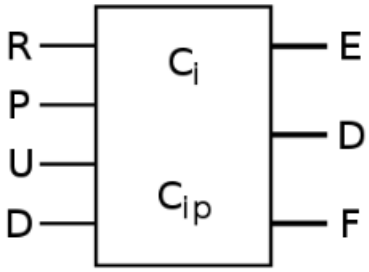
- Entrada UP (U): Cada vegada que s'activa produeix un increment en una unitat de la compta que tingui en aquell moment el comptador.

- Entrada DOWN (D): Cada vegada que s'activa produeix un decrement en una unitat de la compta que tingui en aquell moment el comptador.

- Sortida FULL (F): S'activa al produir-se un desbordament del valor del comptador comptant en sentit ascendent.

- Sortida DONE(D): S'activa quan el valor del comptador es igual al valor preestablert Cip.

- Sortida EMPTY (E): S'activa al produir-se un desbordament del valor del comptador comptant en sentit descendent.



Il·lustració 11. Esquema d'un comptador Ci

2.3.4 SYSMAC STUDIO

Sysmac Studio és la nova plataforma d'automatització per oferir als desenvolupadors de maquinària un control total des d'un entorn únic a través d'una única connexió i un sol programari. Sysmac Studio integra funcions de configuració, programació, simulació i monitoratge des d'un senzill entorn. Aquesta avançada eina de programari és l'única aplicació que necessita per al controlador d'automatització de màquines de la sèrie NJ, que unifica lògica, motion i visió en una única plataforma (Omron).

Sysmac Studio inclou configuració, programació, simulació i monitoratge a més d'una xarxa d'alta velocitat per màquines, EtherCAT, que s'utilitza per controlar el motion, la visió, els sensors i els actuadors.

El principal producte maquinari que integra aquest entorn és (Omron):

- Controladors de la sèrie NJ / NX
- HMI de la sèrie NA
- E/S i seguretat de la sèrie NX
- Servo de la sèrie G5
- Variadors de freqüència de les sèries MX2 i RX
- Sistema de visualització i sensor FH i FQM
- Components de xarxa GX
- Sensors E3 N-Smart
- Sensors de mesurament de la sèrie ZW

El paquet de programari també inclou altres funcions de disseny de HMI i dades addicionals de configuració de la xarxa.

Compleix íntegrament l'estàndard obert IEC 61131-3, proporciona un entorn de programació d'avantguarda basat en diagrama de relés, llenguatges de programació de text estructurats i unitats, funcions, blocs de funcions, tipus de dades i estructures de organització de programes.

Compta amb una simulació que permet netejar i depurar el programa ja sigui parcial o complet, de manera que facilita un desenvolupament més ràpid i redueix el nombre d'errors.

La gestió de tasques és una funció del controlador de la sèrie NJ i de Sysmac Studio que garanteix que el seu sistema de control funciona exactament com està previst. El sistema de tasques utilitzat pels controladors de la sèrie NJ s'utilitza per assignar unes condicions d'execució i una ordre d'execució a una sèrie de processos, com el refresc d'E/S i l'execució de programes de l'usuari. Aquest nou nivell de control permet al programador seleccionar com funcionarà la CPU durant el seu temps d'execució, per garantir un funcionament uniformement ràpid durant tota la vida útil de la màquina. Sysmac Studio proporciona ajustos simples i clars per a la seva configuració de tasques, així com potents eines d'avaluació del rendiment. Durant la selecció de tasques només es mostren les opcions vàlides, orientant l'usuari cap a la configuració adequada. Un cop seleccionada una configuració, pot supervisar el temps del programa per garantir un funcionament òptim. El monitor de temps d'execució de tasques pot utilitzar juntament amb el simulador per mostrar els temps d'execució abans de la descàrrega a el controlador físic. Aquestes noves eines de programari permeten a l'usuari predir el funcionament de la màquina abans

de la implementació, en lloc d'haver de reaccionar davant respostes inesperades durant la posada en marxa (Omron).

2.3.5 KEPSERVEREX I OPC QUICK CLIENT

En el camp de les comunicacions industrials existeixen diversos protocols utilitzats per a l'intercanvi d'informació entre una gran varietat de dispositius industrials com ara PLCs, sensors, robots, i sistemes de supervisió. Un dels protocols de comunicació més importants és l'estàndard OPC.

L'OPC (OLE for Process Control) és un estàndard de comunicació en el camp del control i supervisió de processos industrials, basat en una tecnologia Microsoft, que ofereix una interfície comuna per a comunicació que permet que components programari individuals interactuïn i comparteixin dades. La comunicació OPC es realitza a través d'una arquitectura Client-servidor. El servidor OPC és la font de dades (com un dispositiu maquinari a nivell de planta) i qualsevol aplicació basada en OPC pot accedir a aquest servidor per llegir / escriure qualsevol variable que ofereixi el servidor. (AS, 2017)

KEPServerEX és un programa que facilita la interconnexió de dispositius utilitzant l'estàndard de comunicació OPC, actuant com a vincle entre múltiples dispositius que poden estar presents en entorns industrials.

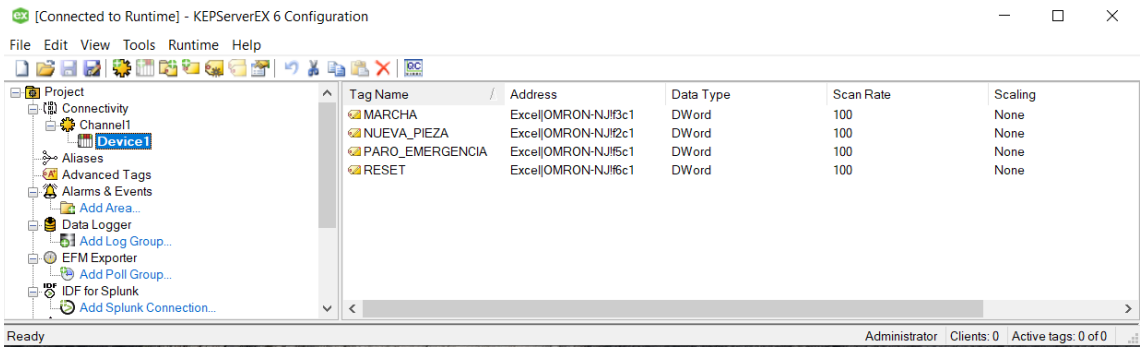
La comunicació per mitjà de KepserverEX, s'estableix amb tres paràmetres que són el canal, el dispositiu, i els ítems. Per assegurar que la connexió OPC és segura, cal crear usuaris i grups d'usuaris que seran exclusius per a aquest ús.

- Channel: El servidor OPC admet l'ús de diversos controladors de comunicacions simultanis. Un projecte de servidor pot constar de molts canals amb el mateix controlador de comunicacions o amb controladors de comunicacions únics. Un canal actua com el component bàsic d'un enllaç OPC.

- Device: Un dispositiu la representació que es dóna al servidor a cada element present en un canal, així un canal configurat per a un protocol pot tenir múltiples elements.

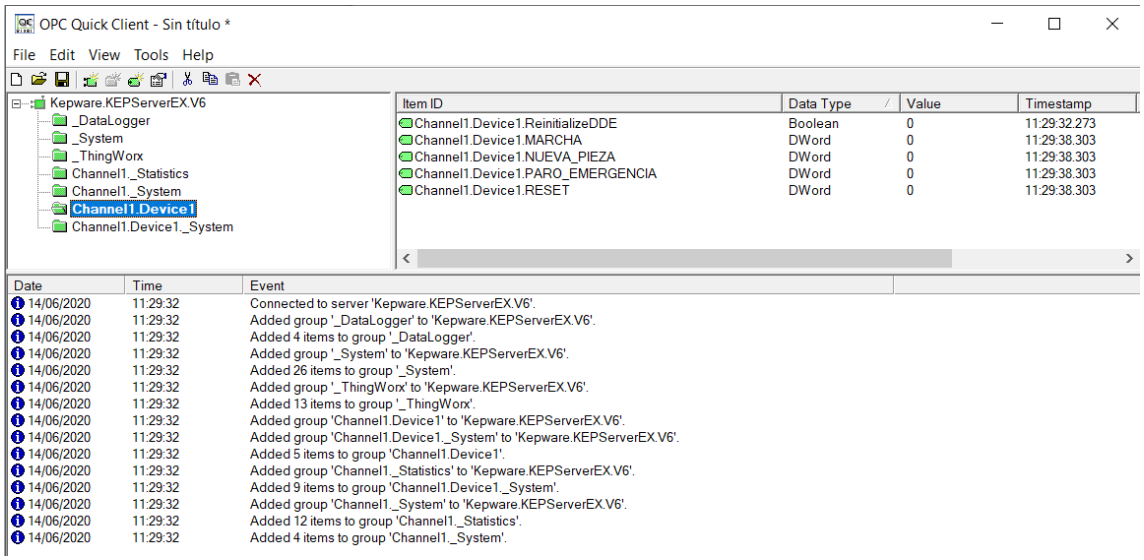
- Ítem: Un ítem és la representació que es dóna a les variables presents en un dispositiu, cada dispositiu pot gestionar diverses variables per al seu funcionament. Per exemple, una variable pot ser la lectura d'estat d'un sensor, que és d'interès conèixer en un sistema de supervisió.

KEPSeverEX és la plataforma líder en connectivitat industrial, proporcionant accés a l'entorn industrial o a la infraestructura a través dels seus més de 150 protocols incorporats i servint aquesta informació a OPC DA i OPC UA. KEPServerEX permet a l'usuari connectar, gestionar, monitoritzar i controlar qualsevol tipus de dispositiu a través d'una única i intuïtiva interfície (Logitek Team, 2019).



Il·lustració 12. Interfície KepServerEX

Per poder executar un bon enllaç client-servidor, es necessari l'OPC Quick client, ja que el seu objectiu es crear clients que puguin tenir accés a servidors OPC per poder interactuar amb ells.



Il·lustració 13. Interfície OPC Quick Client

2.3.6 MICROSOFT EXCEL

Degut a les conseqüències de la pandèmia ocasionada pel SARS-CoV-2 (COVID-19), no s'ha pogut accedir a l'estació II del Laboratori de Robòtica i

CIM, per tant, per simular el funcionament de l'estació automatitzada amb el reconeixement de veu, s'ha decidit utilitzar el software Microsoft Excel.

Microsoft Excel és un full de càlcul desenvolupat per Microsoft per a Windows, MacOS, Android i iOS. Compta amb càlcul, eines de gràfics, taules dinàmiques, i un llenguatge de programació anomenat macro de Visual Basic per a Aplicacions. Ha estat un full de càlcul molt àmpliament aplicat per a aquestes plataformes, sobretot des de la versió 5 el 1993, i s'ha reemplaçat Lotus 1-2-3 com l'estàndard de la indústria per a fulls de càlcul. Excel forma part de la suite de Microsoft Office de programari (Korsan, S.) .

La versió de Windows d'Excel dóna suport a la programació a través de Microsoft Visual Basic per a aplicacions de VBA, que és un dialecte de Visual Basic. Programació amb VBA permet la manipulació de fulls de càlcul que és difícil o impossible amb les tècniques estàndard de fulles electròniques. Els programadors poden escriure codi directament utilitzant l'Editor de Visual Basic VBE, que inclou una finestra per a programar, depuració de codi, i l'entorn codi de l'organització mòdul (Korsan, S.).

L'estació automatitzada ha sigut programada amb visual basic (vba), un llenguatge de programació que a permès executar amb èxit el funcionament seqüencial de l'estació automatitzada.

3. RECONEIXEMENT DE VEU

3.1 INTRODUCCIÓ

En aquest capítol es detalla el desenvolupament de l'aplicació de reconeixement de veu amb l'objectiu de controlar el moviment de l'estació automatitzada. Primerament, s'explica el funcionament teòric del reconeixement de veu. Un cop explicat el funcionament del reconeixement de veu, s'explica la configuració de l'ordinador i el funcionament de l'aplicació de reconeixement de veu amb les funcions i característiques que presenta. Després d'explicar tot el procés de configuració, preparació i funcionament de l'aplicació, es detalla tot el procés de configuració del servidor OPC, la connexió entre el sistema de reconeixement de veu i el servidor OPC.

3.2 HISTÒRIA DEL RECONEIXEMENT DE LA VEU

La història del reconeixement de la veu, es remunta a l'any 1870, on Alexander Graham Bell volia construir un sistema o dispositiu que fes la parla visible a les persones amb problemes auditius. El resultat d'aquest estudi i construcció del dispositiu va ser el telèfon (Cardenas, 2009).

A l'any 1880, Tihamir Nemes, sol·licita el permís per a una patent per desenvolupar un sistema de transcripció automàtica que identifiqués seqüències de sons i els imprimís (text). Però finalment va ser rebutjat com a "Projecte no realista" (Cardenas, 2009).

30 anys després AT % T Bell Laboratoris, va construir la primera màquina capaç de reconèixer veu (basada en plantilles) dels 10 dígits de l'anglès. Requeria extens reajustament a la veu d'una persona, però un cop aconseguit tenia un 99% de certesa. Per tant sorgeix l'esperança que el reconeixement de veu és simple i directe (Cardenas, 2009).

A mitjans dels anys 60, la majoria dels investigadors reconeix que era un procés molt més intricat i subtil del que havien anticipat. Per tant comencen a reduir l'abast i s'enfoquen a sistemes més específics (Cardenas, 2009):

- Dependents del Locutor.
- Flux discret de parla (amb espais / pauses entre paraules)
- Vocabulari petit (menor o igual a 50 paraules)

Aquests sistemes comencen a incorporar tècniques de normalització del temps (minimitzar diferència en velocitat de la parla) i van deixar de buscar una exactitud perfecta al reconeixement (Cardenas, 2009).

Paral·lelament, IBM i CMV treballen en reconeixement de veu continu però no es veuen resultats fins als anys 70 (Cardenas, 2009).

A principis dels anys 70, es produeix el 1er Producte de reconeixement de veu, el VIP100 de Threshold Technology Inc (utilitzava un vocabulari petit, dependent del locutor, i reconeixia paraules discretes). Guanya el U.S. National Award el 1972 (Cardenas, 2009).

Anys més tard, neix l'interès d'ARPA del U.S. Department of Defense, i gràcies al llançament de grans projectes de Recerca i finançament per part del govern es precipita l'època de la intel·ligència artificial. El projecte finançat per ARPA busca el reconeixement de parla contínua, de vocabulari gran. Impulsa que els investigadors s'enfoquin a l'enteniment de la parla. Els sistemes comencen a incorporar mòduls de (Cardenas, 2009):

- Anàlisi lèxic (coneixement lèxic)
- Anàlisi sintàctica (Estructura de Paraules)
- Anàlisi semàntica (significat)
- Anàlisi pragmàtica (Intenció)

Aquests projectes acaben el 1976 amb el resultat que CMU, SRI i MIT van crear sistemes per al projecte ARPA SUD (Speech Understanding Research) (Cardenas, 2009).

Durant els anys 80, el reconeixement de veu es va veure afavorit per tres factors (Ahuactzin, 1999):

- Creixement d'ordinadors personals
- Suport de l'ARPA
- Costs reduïts d'aplicacions comercials

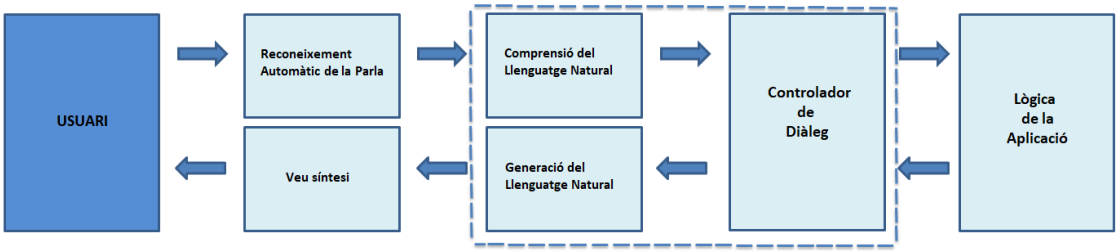
El major interès durant aquest període de temps era el desenvolupament de vocabularis grans. A l'any 1985, un vocabulari de 100 paraules era considerat gran. Tot i així, en 1986 va haver-hi un de 20.000 paraules. Durant aquesta època també van haver-hi grans avenços tecnològics, ja que, es va canviar l'enfocament basat en reconeixement de patrons a mètodes de modelat probabilístics, com els Models Ocults de Markov (HMM) (Ahuactzin, 1999).

Per als anys 90, els costos de les aplicacions de reconeixement de veu van continuar decreixent i els vocabularis extensos van començar a ser normals. També les aplicacions independents del locutor i de flux continu, és a dir, no hi

ha pauses insignificants a la parla) van començar a ser més comuns (Ahuactzin, 1999).

3.3 FUNCIONAMENT

Els sistemes de reconeixement automàtic de la parla són aquells que tradueixen expressions i enunciats de la seva forma parlada a text. Aquest reconeixement no comporta la interpretació i enteniment del significat del text, ja que aquesta tasca és realitzada posteriorment, en cas que l'aplicació el requereixi, mitjançant tècniques de Processament de Llenguatge Natural (Campos, 2018).



Il·lustració 14. Sistema de comunicació verbal humà-màquina (Campos, 2018)

La majoria dels sistemes de reconeixement de parla moderns es basen en el que es coneix com a Model de Markov ocult (HMM). Aquest enfocament funciona amb el supòsit que un senyal de veu, quan es visualitza en un termini prou curt (per exemple, deu mil·lisegons), es pot aproximar raonablement com un procés estacionari, és a dir, un procés en què les propietats estadístiques no canvien amb el pas del temps (Amos, 2018).

En un HMM típic, el senyal de veu està dividit en fragments de 10 mil·lisegons. L'espectre de potència de cada fragment, que és essencialment una trama de la potència del senyal com a funció de freqüència, s'assigna a un vector de nombres reals coneguts com a coeficients ceprals. La dimensió d'aquest vector sol ser petita, de vegades fins a 10, encara que sistemes més precisos poden tenir una dimensió de 32 o més. La sortida final de l'HMM és una seqüència d'aquests vectors (Amos, 2018).

Per descodificar el discurs en text, els grups de vectors s'uneixen a un o diversos fonemes, una unitat fonamental de la parla. Aquest càlcul requereix formació, ja que el so d'un fonema varia d'altaveu a altaveu, i fins i tot varia d'una declaració a una altra del mateix altaveu. Aleshores s'aplica un algorisme especial per determinar la paraula (o paraules) més probable que produeixi la seqüència donada de fonemes (Amos, 2018).

Es pot imaginar que tot aquest procés pot ser costós computacional. En molts sistemes moderns de reconeixement de la parla, les xarxes neuronals s'utilitzen per simplificar el senyal de veu utilitzant tècniques per a la transformació de funcions i la reducció de la dimensionalitat abans del reconeixement HMM. Els detectors d'activitat de veu (VADs) també s'utilitzen per reduir un senyal d'àudio a només les porcions que probablement continguin veu. D'aquesta manera, s'evita que el reconeixement perdi el temps que analitza parts innecessàries del senyal (Amos, 2018).

3.3.1 PROCÉS DE RECONEIXEMENT

La unitat fonamental del so parlat és el fonema i a la seva realització física se'l coneix com "fono". A un fonema li poden correspondre diversos "fonos" equivalents denominats al·lòfons, ja que la seva articulació pot variar depenent dels fonemes circumdants.
Per exemple el fonema /d/ en espanyol compta amb almenys dos al·lòfons: La pronunciació oclusiva (que impedeix la sortida de l'aire) [d] present en la paraula tendal (Campos, 2018).

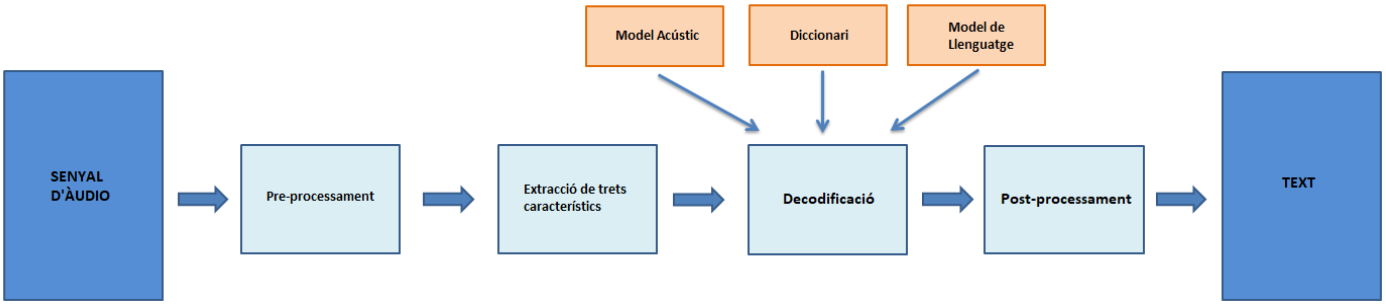
La pronunciació fricativa (que permet la sortida de l'aire pels costats) [đ] trobada en la paraula tot (Campos, 2018).

Les paraules estan conformades per un o més fonemes en seqüència, que en ser emesos per un humà, poden ser gravats com un senyal acústic continu; sent l'objectiu d'un sistema de reconeixement de la parla l'inferir les paraules originals pronunciades per l'humà a partir d'aquest senyal (Campos, 2018).

Per a aconseguir l'anterior, comunament s'utilitza un enfocament probabilístic en el qual el senyal parlat correspon a una seqüència de paraules amb certa probabilitat. En avaluar diferents seqüències de paraules d'acord amb les propietats fonètiques del senyal i al coneixement lingüístic sobre la contigüitat de les paraules, així com la gramàtica per a la correcta formació de les frases, es dóna una puntuació a cada frase candidata, triant-se com a resultat aquella que tingui la millor puntuació (Campos, 2018).

L'esquema tradicional d'un sistema típic de reconeixement de la parla consisteix en els següents passos (Campos, 2018):

- Pre-processament (inclou segmentació del senyal).
- Extracció de trets característics.
- Descodificació, emprant diccionaris amb models acústics i de llenguatge.
- Post-processament del resultat.



Il·lustració 15. Esquema d'un sistema de reconeixement de la parla (Campos, 2018)

3.3.2 PRE-PROCESSAMENT

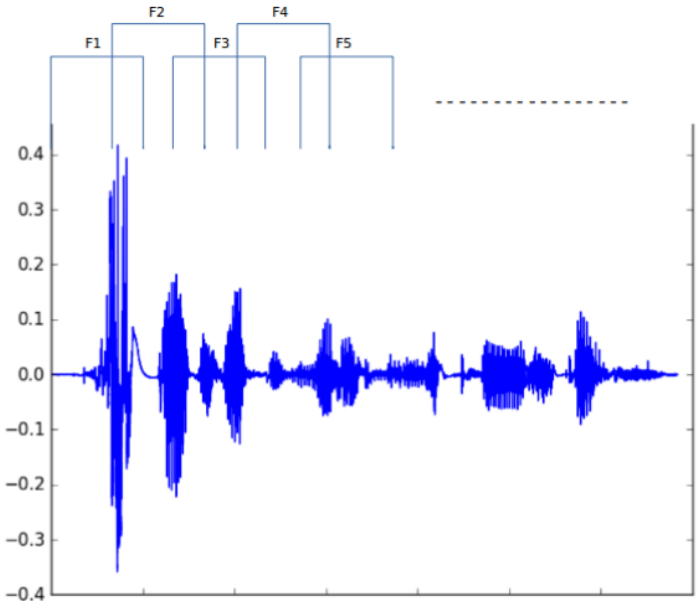
El senyal de parla és una mesura observada, realitzada respecte al pas del temps. Pot veure's com el mapatge de la força de les ones de so de la veu en un determinat moment del temps. És important tenir en compte que aquest valor no es relaciona únicament amb alguna ona de freqüència específica (Campos, 2018).

Per a simplificar el processament de senyals continus, el conjunt infinit de valors possibles que el sistema continu pot prendre en un interval ha de reduir-se a un conjunt finit d'interval [a, b], a través d'un altre procés de mapatge anomenat mostreig. Aquesta acció es diu discretització i el senyal recentment definit, capaç d'assignar aquest conjunt finit de punts a un mesurament de nivell superior, es diu senyal discret. Per descomptat, en general, també es pot imposar una restricció similar al rang del mapatge i reduir-lo a un conjunt finit, però això no és necessari (Campos, 2018).

La senyal sonora és gravada i discretitzada amb certa freqüència (usualment 16 kHz encara que en línies telefòniques la freqüència normalment està limitada a 8 kHz) i se li apliquen certs filtres per a reduir el soroll de fons i magnificar les diferències entre diferents tipus de vocalització (Campos, 2018).

3.3.3 EXTRACCIÓ DE TRETS CARACTERÍSTICS

Per a extreure trets es divideix el senyal en finestres de temps usualment de 25 ms cadascuna amb una superposició de 10 ms entre elles. Al senyal acústic dins de cada finestra se li apliquen certes transformacions matemàtiques, com Fourier i cosinus discret, a més d'altres filtres i processos de normalització per a obtenir un vector de coeficients representatiu del senyal (Campos, 2018).



Il·lustració 16. Divisió de la senyal acústica en finestres de temps superposades (Campos, 2018)

3.3.4 DECODIFICACIÓ

En aquesta etapa es calcula com és la seqüència de paraules més probable de correspondre al senyal representat pels vectors de trets característics. Per a això es consideren tres diferents fonts d'informació (Campos, 2018):

- Un model acústic, típicament un Model Ocult de Markov associat a cada fonema o paraula.
- Un diccionari, consistent en una llista de paraules i els fonemes que les conformen.
- Un model de llenguatge, amb probabilitats de paraules i seqüències d'elles.

Prenent els vectors de trets característics com les observacions O, es busca la seqüència de paraules W que maximitzi la probabilitat a posteriori P(W|O), la qual cosa matemàticament es representa de la següent forma (Campos, 2018):

$$\widehat{W} = \operatorname{argmax}_w P(W|O)$$

Utilitzant la regla de Bayes per a resoldre l'equació s'obté (Campos, 2018):

$$P(W|O) = \frac{P(O|W)P(W)}{P(O)}$$

Si s'ignora el denominador comú per a totes les observacions, s'obté l'equació fonamental del reconeixement de la parla (Campos, 2018):

$$\widehat{W} = \text{argmax}_w P(O|W)P(W)$$

L'equació anterior té com a components el model acústic P(O|W) que descriu la distribució de les observacions donada una seqüència de paraules i el model del llenguatge P(W) basat únicament en la seqüència de paraules. La seqüència de paraules que resulti amb major puntuació és aquella que es postula com la transcripció de la parla estimada (Campos, 2018).

3.3.5 POST-PROCESSAMENT

El resultat de la descodificació no ha de limitar-se a seleccionar la frase amb millor puntuació com el resultat definitiu, sinó que pot produir una llista amb les millors frases candidates. Durant el post-processament poden realitzar-se ajustos emprant fonts addicionals d'informació específica a cert context o utilitzant algorismes més precisos però costosos, que modifiquin la hipòtesi final (Campos, 2018).

3.4 CONFIGURACIÓ AMB WINDOWS 10

El reconeixement de veu es durà a terme en un ordinador que compta amb una arquitectura de 64 bits, per tant tots els programes, llibreries i tots els arxius necessaris preferiblement hauran de ser compatibles amb aquesta arquitectura. S'ha de tenir en compta que un ordinador amb una arquitectura de 64 bits també suporta programari de 32 bits.

De fet, es va intentar realitzar amb la versió 3.8.3 de 64 bits de Python, però a l'hora de connectar el client Python amb el servidor OPC KEPServer, no s' aconseguia establir la connexió. Per tant, després d'investigar d'on podia sorgir aquest problema, es va decidir provar amb la mateixa versió 3.8.3 però amb arquitectura de 32 bits i es va poder establir connexió amb el servidor OPC KEPServer perfectament, per tant, la versió de Python utilitzada en aquest projecte és Python 3.8.3.

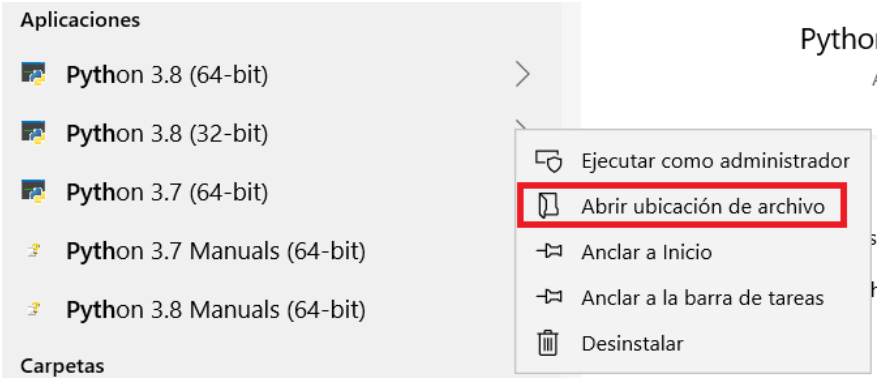
3.4.1 PYTHON 3.8.3

En aquest apartat s'explica com utilitzar correctament el software Python, de manera que primer de tot, s'haurà de descarregar el programa a la següent web on trobarem la versió més actual:

https://www.python.org/downloads/

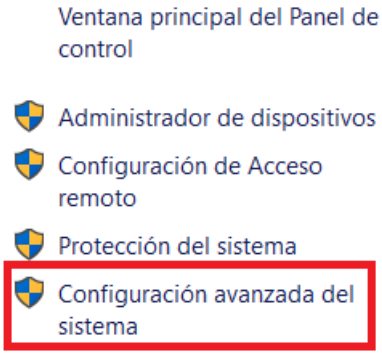
Un cop descarregat l'arxiu l'hem d'executar com un administrador de manera que al fer la instal·lació es concedeixin tots els permisos.

Un cop s'ha instal·lat correctament l'arxiu és important trobar on està instal·lat Python per poder copiar la ruta i guardar-la, ja que s'haurà d'utilitzar per poder preparar el sistema per executar programes escrits en aquest llenguatge de programació. Els passos a seguir són els següents:



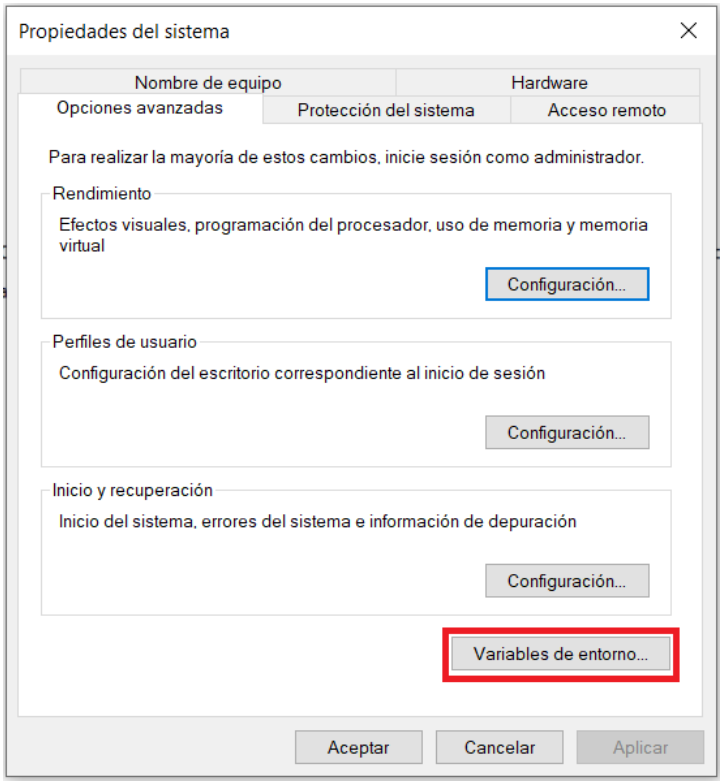
Il·lustració 17. Direcció d'instal·lació Python 3.8

A l'explorador de Windows, s'ha de prémer el botó dret a sobre "El meu equip" i seleccionar "propietats". Un cop obertes les opcions del sistema s'ha de seleccionar "Configuració avançada del sistema":



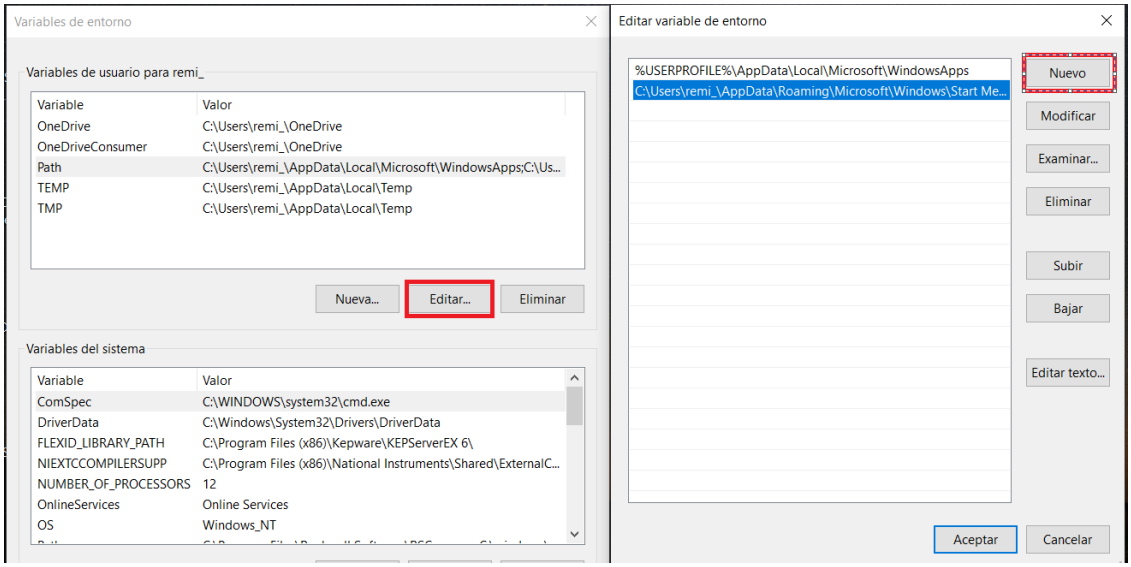
Il·lustració 18. Configuració avançada del sistema

Prement la opció de "Configuració avançada del sistema" s'obre la finestra de propietats del sistema en la qual s'ha de seleccionar "Opcions avançades" i després "Variables d'entorn":



Il·lustració 19. Propietats del sistema

En aquesta finestra que s'obre s'ha de seleccionar la variable "Path" i prémer el botó "Editar", per poder crear i col·locar una nova direcció on es troba instal·lat el programa Python. Un cop afegida la ruta que prèviament hem hagut de guardar, es podran executar programes de Python des de la consola de Windows:



Il·lustració 20. Edició de les variables d'entorn

3.4.2 LLIBRERIES PYTHON

A continuació, es mostraran les diferents llibreries que es recomana tenir instal·lades per poder dur a terme el procés del reconeixement de veu, com poden ser llibreria SpeechRecognition, llibreria PyAudio i llibreria pip, així com la configuració de l'API de Google Speech to text, utilitzada per aquest projecte.

3.4.3 LLIBRERIA PIP

Pip de Python és un sistema de gestió de paquets utilitzats per instal·lar i administrar paquets de software escrits en Python. Per poder instal·lar aquesta llibreria es necessari executar la comanda des de la consola de Windows.

Generalment, la llibreria pip està instal·lada per defecte al instal·lar Python, però hi ha alguns casos en els que no funciona correctament, per tant, es necessari instal·lar-la manualment. Per a la correcta instal·lació de la llibreria pip s'ha d'escriure la següent comanda a la consola de Windows:

```
>> python.get – pip.py
```

Un cop està instal·lada la llibreria pip, serà necessari actualitzar-la per poder obtenir la versió més actual. Per poder actualitzar la llibreria recentment instal·lada s'ha d'escriure la següent comanda:

```
>> python – m pip install – U pip
```

D'aquesta manera s'aconsegueix tenir la llibreria pip actualitzada, això permet instal·lar nous mòduls en Python executant comandes des de la consola de Windows. Els mòduls que s'instal·lin han d'estar indexats als paquets de Python.

3.4.4 LLIBRERIA PYAUDIO

PyAudio proporciona enllaços de Python per a PortAudio, la biblioteca de E/S d'àudio multi-plataforma. Amb PyAudio, es pot utilitzar Python per reproduir i gravar àudio en una varietat de plataformes, per tant, és una llibreria bàsica per poder realitzar el reconeixement de veu de la estació.

Per poder instal·lar PyAudio s'ha d'executar la següent comanda a la consola de Windows:

```
>> pip install PyAudio
```

En alguns casos la funció d'instal·lació de PyAudio no es pot executar des de la consola de Windows. Quan això passa, s'han d'instal·lar els arxius Wheel manualment, per tant, el que s'ha de fer és descarregar-se l'arxiu corresponent a la versió de Python i Windows de l'ordinador on es vol instal·lar. En el meu cas, vaig trobar els arxius Wheel en el següent link:

https://www.lfd.uci.edu/~gohlke/pythonlibs/#pyaudio

L'arxiu a descarregar és el següent:

PyAudio-0.2.11-cp37-cp37m-win_amd64.whl

Un cop descarregat l'arxiu per a la versió Windows utilitzada en aquest projecte, aquest arxiu s'ha d'ubicar en un directori i executar la següent comanda en la consola de Windows:

>> pip install PyAudio-0.2.11-cp37-cp37m-win_amd64.whl

Un cop instal·lada aquesta llibreria, qualsevol programa creat amb Python serà capaç d'accedir al micròfon i realitzar gravacions de so, les quals seran les instruccions de veu necessàries per l'estació automatitzada.

3.4.5 LLIBRERIA SPEECH RECOGNITION

El reconeixement de veu és el procés de convertir paraules parlades en text. Gràcies al reconeixement de veu s'obtenen les instruccions que finalment seran necessàries per al moviment de l'estació.

La llibreria SpeechRecognition és una biblioteca que treballa pel reconeixement de veu i en la qual hi ha un bon nombre de paquets que ajuden a facilitar aquest procés d'obtenció del reconeixement de la parla. Aquests paquets són (Amos, 2018):

- apiai
- assemblyai
- google-cloud-speech
- pocketsphinx
- SpeechRecognition
- watson-developer-cloud
- wit

Alguns d'aquests paquets, com ara wit i apiai, ofereixen funcions integrades, com el processament del llenguatge natural per identificar la intenció d'un parlant, que van més enllà del reconeixement bàsic de la parla. Altres, com ara google-cloud-speech, es centren exclusivament en la conversió de veu a text (Amos, 2018).

Per a aquest projecte s'ha decidit utilitzar l'API de google-cloud-speech, ja que, segons les seves característiques s'ajusta perfectament al que es busca per poder realitzar amb èxit aquest projecte. A més, apart d'ésser un motor de reconeixement de veu que té molta integració amb altres serveis i està en constant desenvolupament, té suport en molts idiomes i moltes funcions per poder millorar la qualitat d'àudio obtingut pel micròfon i no requereix de cap clau d'API.

Per utilitzar tota la funcionalitat de la biblioteca, s'haurà de tenir (Zhang, 2017):

- Python 2.6, 2.7 o 3.3+ (Obligatori)
- PyAudio 0.2.11+ (Necessari només si s'ha d'introduir micròfon)
- PocketSphinx (Necessari només si es necessita utilitzar el reconegut Sphinx, recognizer_instance.recognize_sphinx)
- Biblioteca de clients de l'API de Google per a Python (Necessària només si s'ha d'utilitzar l'API de Google Cloud Speech, recognizer_instance.recognize_google_cloud)
- Codificador FLAC (Necessari només si el sistema no està basat en x86 Windows / Linux / OS X)

Per poder instal·lar la llibreria SpeechRecognition, es pot executar la següent comanda en la consola de Windows:

```
>> pip install SpeechRecognition
```

Un cop instal·lat aquest mòdul, el programa ja serà capaç de poder convertir a text les instruccions de veu que rebi de l'arxiu d'àudio.

3.4.6 CONFIGURACIÓ API DE GOOGLE SPEECH TO TEXT

Com s'ha esmentat anteriorment, en el projecte s'utilitza l'API de reconeixement de veu de Google, Speech to text. El funcionament d'aquesta API és el següent:

Primerament, s'obté un arxiu d'àudio que s'ha obtingut mitjançant el micròfon, després Python puja l'arxiu d'àudio obtingut a processar al servidor de Google Cloud, el qual dóna la transcripció del que s'ha escoltat.

Per poder interpretar la informació obtinguda serà necessari la realització de les següents accions:

- S'ha d'importar la llibreria de reconeixement de veu a Python:

```
import speech_recognition as sr
```

- S'ha de crear una funció que serà l'encarregada de gravar l'àudio a través del micròfon:

```
audio = r.listen(source) #timeout=5,phrase_time_limit=10)
```

- S'ha de crear una funció de lectura, on s'ajusta la sensibilitat del micròfon:

```
with sr.Microphone() as source:
    print('Quina acció desitges realitzar?')
    r.adjust_for_ambient_noise(source)
    audio = r.listen(source) #timeout=5,phrase_time_limit=10)
try:
```

- S'ha de crea la instància "r", que és l'encarregada de l'ús de la funció del reconeixement de veu:

```
r = sr.Recognizer()
```

- S'ha de configurar l'ús de l'API de Google:

```
try:
    respuesta = r.recognize_google(audio,language='es-ES, en-US, en-GB')
    recognized = r.recognize_google(audio,show_all=True,language='es-ES, en-US, en-GB')

except sr.RequestError:
    print('Aplicació no disponible')

except sr.UnknownValueError:
    print('Incapaç de reconèixer el discurs')
```

- S'han de crear unes funcions (respuesta, recognized) encarregades de rebre la transcripció de l'àudio generat:

```
print('Es realitzarà la següent acció:', recognized)
print (respuesta)
print
```

Amb aquestes funcions i configuracions, un cop el micròfon comença a gravar, aquest no s'atura fins que hi hagi una pausa transcrivint així tot el que s'hagi parlat fins al moment. Generalment el servidor té un "time out" de 5 segons.

3.4.7 ENTRENAMENT DE LES INSTRUCCIONS AMB RECONEIXEMENT DE VEU

Generalment, hi ha diversos factors que poden afectar al desenvolupament de la funció de reconeixement de veu en el moment de la gravació i creació de l'arxiu d'àudio.

Primerament per l'idioma, en aquest projecte la majoria d'instruccions són en castellà, però també n'hi ha una en anglès. També pot existir soroll ambiental que pot distorsionar la gravació. Per això, és molt important veure la transcripció de manera que es pugui veure si l'àudio concorda amb la transcripció feta.

Degut a que a vegades algunes transcripcions per part de l'API de Google que no concorden amb la instrucció demanada, s'ha decidit entrenar cadascuna de les instruccions per tal de recollir el màxim de variacions similars o expressions que s'assemblin a la instrucció demanada.

El procés d'entrenament de les instruccions és el següent:

- Acció a realitzar per l'estació automatitzada
- Buscar una instrucció simple i ràpida per aquesta acció a realitzar
- Buscar paraules similars a la instrucció desitjada
- Realitzar una prova amb el sistema de reconeixement de veu pronunciant la instrucció i totes les paraules similars trobades, d'aquesta manera es pot arribar a més expressions similars que abans no s'havien tingut en compte i que l'API de Google també pot reconèixer fàcilment com la instrucció principal.
- Guardar els resultats obtinguts amb cadascuna de les variacions trobades, ja que aquesta llista de paraules anirà al codi del programa de Python.

A continuació, la taula 2 mostra totes les instruccions a realitzar per l'estació automatitzada:

INSTRUCCIÓ	FUNCIÓ
Marcha	Inicia el procés seqüencial de l'estació
Paro de Emergencia	El sistema es queda totalment parat, esperant l'ordre de Reset per tornar a les condicions inicials.
Reset	Instrucció amb la qual es retorna a les condicions inicials del sistema.
Nueva Pieza	Indica que està entrant una nova peça per la cinta transportadora d'entrada.
Terminar	En quant acabi l'última peça que està entrant o que ja hi és en el procés del sistema, aquest no farà cap peça més fins rebre un altre cop la instrucció 'Marcha'.
Salir	El sistema de control a través del reconeixement de veu s'acaba tancant l'aplicació amb Python.

Taula 2. Instruccions a realitzar per l'estació automatitzada

Aquestes instruccions són les que han sigut entrenades de manera que s'ha guardat un llistat de paraules similars. A la taula 3 es mostren les variacions principals de les instruccions realitzades:

INSTRUCCIÓ	VARIACIONS SIMILARS						
Marcha	Marcha	Martxa	Mancha	Hacha	Marta	Maca	Maka
Paro de Emergencia	Paro de Emergencia	Paro Emergencia	Emergencia	Emergencias	--	--	--
Reset	Reset	Recet	Recette	Receta	Receptor	Raclette	--
Nueva Pieza	Nueva Pieza	Pieza	Empieza	Nueva Pista	Nueva Vista	Cieza	--
Terminar	Terminar	Terminal	Termita	Terminar	--	--	--
Salir	Salir	--	--	--	--	--	--

Taula 3. Variacions similars de les instruccions a realitzar

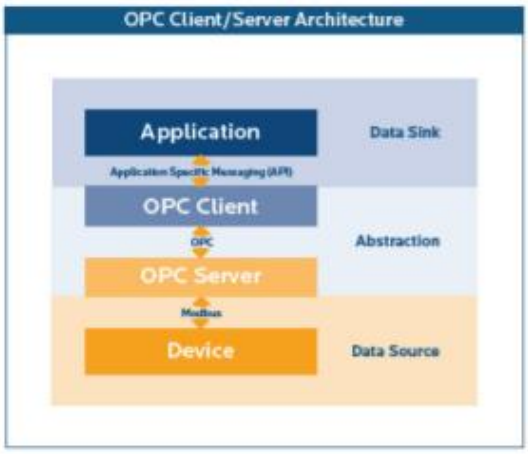
3.5 COMUNICACIÓ OPC

En aquest projecte s'utilitza el programa KEPServerEX, un programa que facilita la interconnexió de dispositius utilitzant l'estàndard de comunicació OPC, actuant com a vincle entre múltiples dispositius que poden estar presents en entorns industrials. En aquest cas, connecta un PLC Omron NJ amb un ordinador on s'executa l'aplicació de reconeixement de veu desenvolupat amb el llenguatge de programació Python.

3.5.1 TECNOLOGIA OPC

OPC és el mètode de connectivitat de dades basat en els estàndards més populars del món. És utilitzat per respondre a un dels majors reptes de la indústria de l'automatització: com comunicar dispositius, controladors i aplicacions sense caure als problemes habituals de les connexions basades en protocols propietaris (Kominek, 2009).

Es pot representar com una capa d'abstracció intermitja que se situa entre la Font de Dades i el Client de dades, permetent intercanviar dades sense saber res un del l'altre. L'abstracció de dispositiu OPC s'aconsegueix utilitzant dos components OPC especialitzats, Client OPC i Servidor OPC (Kominek, 2009).



Il·lustració 21. Arquitectura Client/Servidor OPC (Kominek, 2009)

Un servidor OPC és una aplicació de software desenvolupat específicament per complir amb una o més especificacions OPC. Els servidors OPC són connectors que es poden assimilar a traductors entre el món OPC i els protocols natius d'una Font de Dades. La relació Servidor OPC/Client OPC és de tipus mestre/esclau, per tant, significa que un Servidor OPC només transferirà dades d'una Font de Dades si un Client OPC li demana (Kominek, 2009).

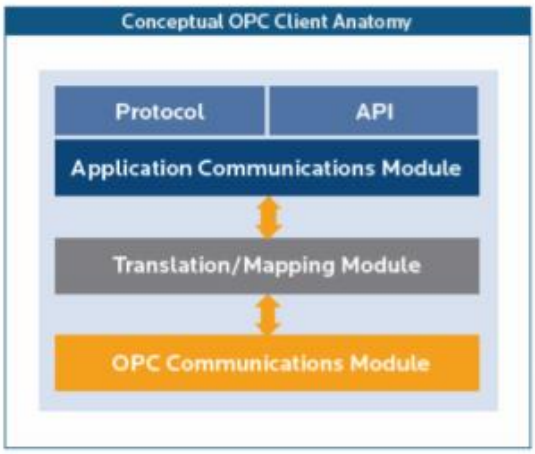
Existeixen cinc tipus de servidors OPC:

- OPC DDE
- Servidor OPC DA
- Servidor OPC AE
- Servidor OPC HDA
- Servidor OPC UA

Un Client OPC és una peça de software creada per comunicar amb Servidors OPC. Representa un destí de dades. Inicien i controlen la comunicació amb Servidors OPC basats en peticions rebudes des de l'aplicació donada en la

petició OPC equivalent i l'envien al servidor OPC adequat per processar la petició (Kominek, 2009).

Els Clients OPC són mòduls de software utilitzats per una aplicació per permetre-la comunicar-se amb qualsevol Servidor OPC compatible visible a la xarxa. Típicament, els Clients OPC estan submergits en aplicacions com HMIs, SCADAs, generadors d'informes, historiadors, convertint-los en aplicacions compatibles OPC (Kominek, 2009).



Il·lustració 22. Anatomia conceptual Client OPC (Kominek, 2009)

3.5.2 OPC DDE

DDE (Dynamic Data Exchange) és un sistema de comunicació inter-processat que permet comunicar o compartir dades entre les aplicacions en sistemes operatius com Windows. Dynamic Data Exchange utilitza la memòria compartida i un conjunt d'ordres, formats de missatges i protocols per a la comunicació i l'ús compartit. L'enllaç i la incrustació d'objectes han tingut èxit a l'intercanvi de dades dinàmic, però aquest últim encara s'utilitza per a tasques de comunicació senzilla i fàcil d'interprocessar (Rouse, 2010).

Als sistemes operatius Windows, OS / 2 i (amb kits de desenvolupament de tercers), Dynamic Data Exchange (DDE) permet compartir o comunicar informació entre els programes. Per exemple, quan es canvia un formulari al programa de base de dades o un element de dades d'un programa de full de càlcul, es poden configurar per canviar també aquests formularis o ítems on es produeixin en altres programes que es poden utilitzar (Rouse, 2010).

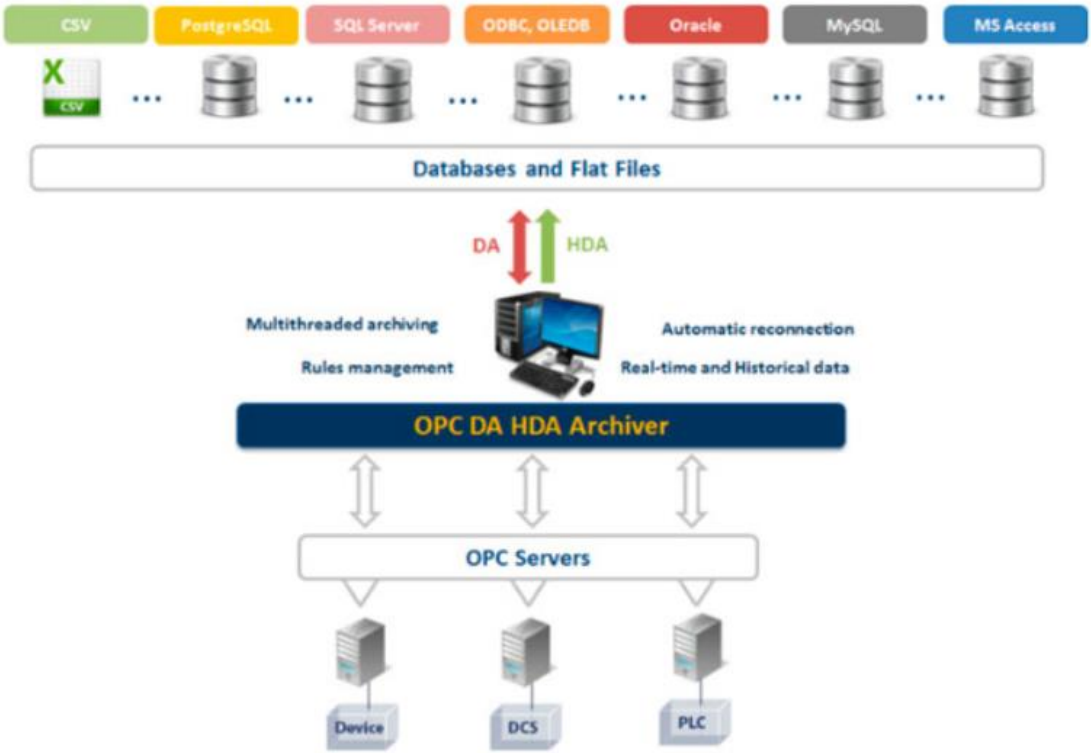
DDE utilitza un model client / servidor en el qual l'aplicació que sol·licita dades es considera el client i l'aplicació que proporciona dades es considera el servidor. Com que es tracta d'un protocol basat en missatges, Dynamic Data Exchange no fa ús de cap biblioteca ni cap funció (Rouse, 2010).

En el cas d'aquest projecte, l'aplicació KEPServerEX per a DDE proveeix connectivitat entre un client OPC i un Servidor DDE. Aquesta aplicació permet llegir i escriure qualsevol variable DDE. Avui dia hi ha milers d'aplicacions que utilitzen DDE, com per exemple, Microsoft Excel, Microsoft Word, Lotus 1-2-3, AmiPro, Quattro Pro i Visual Basic.

3.5.3 OPC DA

OPC DA (Data Access) és una especificació que permet la comunicació i transferència de dades entre una font i una aplicació client, per exemple entre un PLC i un SCADA sense necessitat de que cadascun conegui el protocol natiu de l'altre.

A continuació es mostraran les característiques principals del servidor OPC DA (Cursos Ingenieros Industriales, 2018):



Il·lustració 23. Servidor OPC DA (Cursos Ingenieros Industriales, 2018)

- Permet establir connexió amb qualsevol tipus de dispositiu en temps real, cosa que suposa un gran plus ja que actualment tot canvia pràcticament cada segon.
- OPC DA no requereix de divers propietaris, de manera que s'eliminen tots els problemes i inconvenients d'aquests.

- L'aplicació Client pot tant llegir com escriure dades sense necessitat de conèixer el protocol natiu ni l'estructura de dades del dispositiu al que s'ha de connectar.

3.5.4 OPC AE

OPC AE (Alarms and Events) s'utilitza per intercanviar alarmes i esdeveniments de processos. Proveeix de interfícies on Clients OPC son notificats de successos. Aquests mecanismes es defineixen com:

- Alarma: Condició anormal d'un sistema.
- Condició: Estat nombrat esdeveniment per contenir condicions associades a una etiqueta com pot ser HighAlarm, Normal, LowAlarm.
- Esdeveniment: Ocurrència perceptible, d'importància al servidor OPC, dels dispositius que representa o dels seus dispositius OPC.

3.5.5 OPC HDA

OPC HDA (Historical Data Access) s'utilitza per recuperar i analitzar dades de processos històrics amb el propòsit d'anàlisi, optimització, control d'inventari, compliment normatiu, etc.

Això permet que les aplicacions escriguin dades històriques en una base de dades, ja que molts cops els usuaris necessiten ingressar lectures, o fer correccions d'informació prèvies.

OPC HDA també es pot utilitzar per a la transferència automàtica de dades.

3.5.6 OPC UA

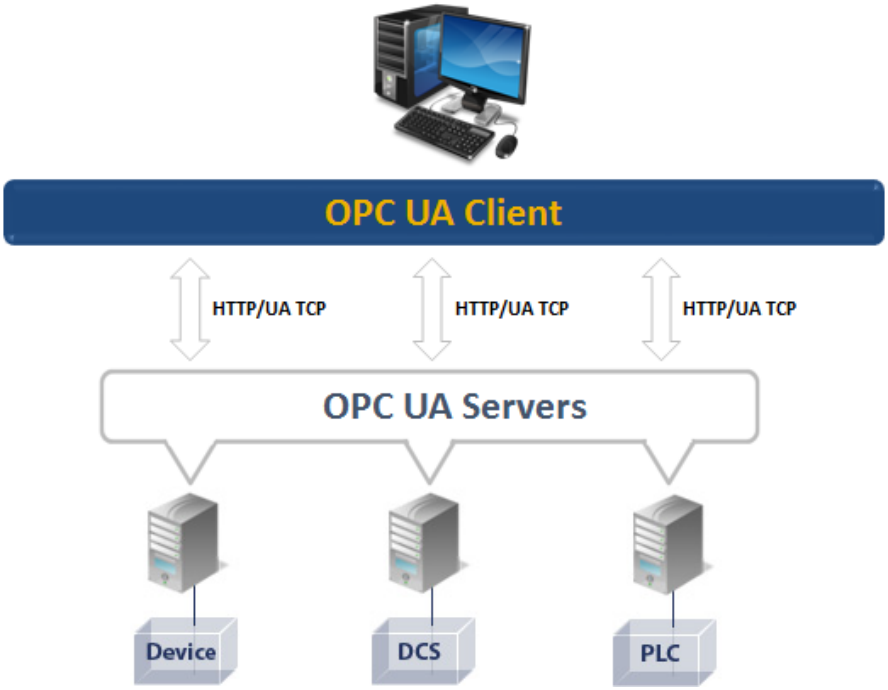
OPC UA (Open Productivity Collaboration Unified Architecture) és l'evolució de l'OPC clàssic, es tracta d'una nova versió d'OPC que és completament compatible amb l'OPC clàssic però va molt més enllà (Cursos de Ingeniería Industrial, 2018).

OPC UA compta amb dos grans característiques, les quals a la vegada, ho diferencien del protocol OPC clàssic. Reemplaça el protocol COM i DCOM, específic de Windows, per uns altres protocols oberts i independents per a que puguin ser utilitzats amb altres sistemes operatius, com per exemple, Linux (Cursos de Ingeniería Industrial, 2018).

Aquest servidor OPC tracta d'interoperabilitat i estandardització. Mentre que OPC convencional resol els problemes d'interoperabilitat a nivell de sistemes

de control de processos, la demanda pel mateix nivell d'estandardització ha estat requerida per l'àrea d'anàlisi de la informació. L'estàndard OPC clàssic està basat en Microsoft DCOM el qual introdueix vulnerabilitat a totes aquestes àrees.

La necessitat de trobar simplicitat, màxima interoperabilitat i seguretat ha portat a la OPC Foundation a la creació d'un mètode de comunicació unificat per a les actuals especificacions OPC DA, HDA, A & E, i Seguretat. OPC UA estén el gran èxit del protocol de comunicació OPC, per a l'adquisició de dades, el modelatge de la informació i la comunicació entre planta i aplicacions d'una manera fiable i segura.



Il·lustració 24. Servidor OPC UA

3.6 CONFIGURACIÓ DEL SISTEMA DE RECONEIXEMENT DE VEU AMB PYTHON

En aquest apartat es detallen tots els requeriments necessaris per poder establir correctament connexió amb un Client OPC amb llenguatge de programació Python. Els requeriments necessaris són:

- Python 3.8.3 32 bits
- PYWIN 32 bits
- OPC DA Auto Wrapper
- OpenOPC per a Python

3.6.1 PYTHON

Com s'ha esmentat anteriorment en l'apartat 2.3.1 del projecte, Python serà el llenguatge de programació que s'utilitzarà per poder accedir a les variables del programa Sysmac Studio. Encara que, degut a la pandèmia ocasionada pel COVID-19, a part de poder accedir a les variables de Sysmac Studio, també s'haurà d'accedir a les variables que té la simulació generada de l'estació automatitzada a Microsoft Excel.

Per poder realitzar això, el primer pas es descarregar-se i instal·lar-se Python 3.8.3 32 bits.

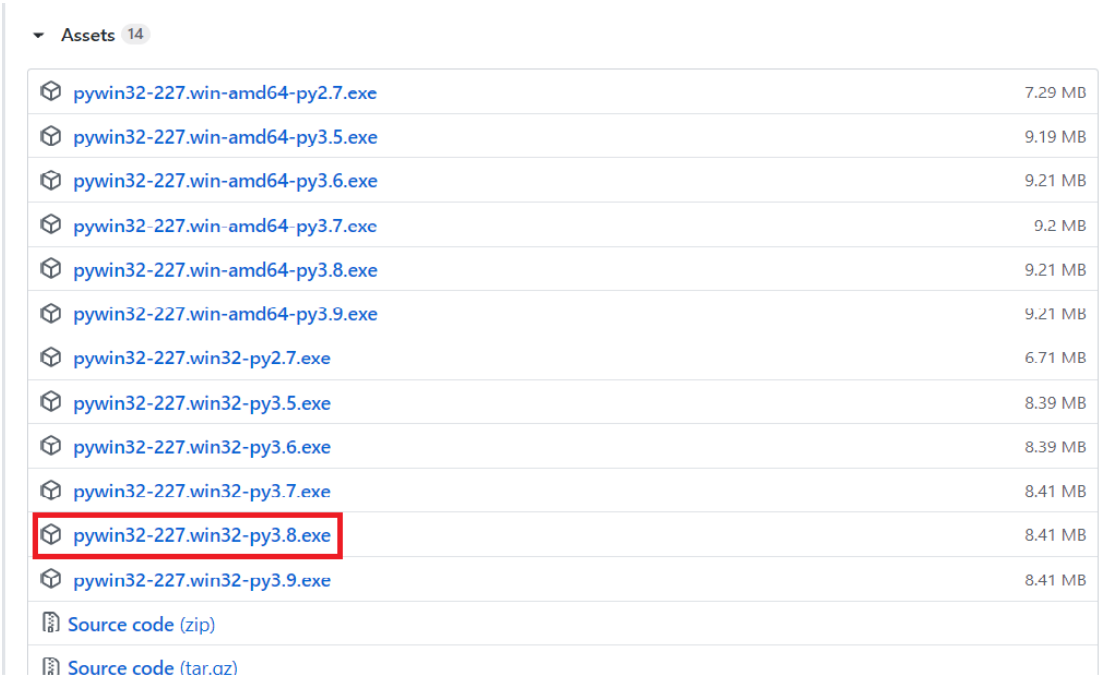
3.6.2 PYWIN 32 BITS

Python té el paquet "Extensions de Python per a Windows" conegut com PYWIN32 que ens permet accedir fàcilment al Model d'Objectes Components (COM) de Windows i controlar les aplicacions de Microsoft a través de Python (Moffitt, 2018).

COM és un sistema independent i distribuït orientat a objectes independent de la plataforma per crear components de programari binaris que puguin interactuar. COM és la tecnologia fonamental de les tecnologies OLE de Microsoft (documents compostos) i ActiveX (components connectats a Internet). Els objectes COM es poden crear amb diversos llenguatges de programació (Batchelor, 2018).

És, per tant, important també descarregar i instal·lar l'extensió de Python PYWIN32. Aquesta extensió proveeix accés a moltes de les APIs de Windows des de Python. Sempre és important descarregar la opció compatible amb la versió de Python. Aquesta extensió es pot descarregar en el següent enllaç:

https://github.com/mhammond/pywin32/releases



Il·lustració 25. Arxiu d'instal·lació de PYWIN32

Un cop instal·lada l'extensió PYWIN32, s'haurà d'executar la següent comanda a la consola de Windows:

>> pip3 install pywin32

Amb tot instal·lat, l'usuari serà capaç de poder accedir fàcilment al Model d'Objectes Components (COM) de Windows i controlar les aplicacions de Microsoft a través de Python.

3.6.3 OPC DA AUTO WRAPPER

Graybox OPC DA Auto Wrapper és un mòdul DLL en el qual s'implementen tots els objectes OLE necessaris. Després de registrar aquest mòdul, es podrà utilitzar qualsevol servidor d'accés a dades de l'OPC amb gairebé qualsevol llenguatge de programació OLE activat (Visual Basic, VBA, etc.) (Graybox Software Developers).

L'objectiu fonamental del disseny és que aquesta interfície estigui destinada a funcionar com a "embolcall" per als servidors d'interfície personalitzada d'accés a dades OPC que proporcionin un mecanisme que faciliti l'automatització per a la funcionalitat proporcionada per la interfície personalitzada (Graybox Software Developers).

Els passos per poder instal·lar OPC DA Auto Wrapper són els següents (Graybox Software Developers):

- Descarregar Graybox OPC DA Auto Wrapper amb el següent link:

  http://gray-box.net/daawrapper.php?lang=en

- Copiar l'arxiu gbda_aut.dll a la carpeta Windows\System32
- Executar la següent comanda a la Consola de Windows per registrar el mòdul:

  >> regsvr32 gbda_aut.dll

Finalment, apareixerà una finestra indicant que s'ha insta·lat l'arxiu dll correctament.

3.6.4 OPEN OPC

OpenOPC per a Python és un conjunt d'eines OPC (OLE per al control de processos) de codi obert dissenyat per al seu ús amb el llenguatge de programació Python. Les característiques que el diferencien dels molts equips d'eines d'OPC disponibles són les següents (SourceForge Developers):

- És una eina fàcil d'utilitzar, ja que la biblioteca OpenOPC implementa un nombre mínim de funcions Python que es poden encadenar de diverses maneres, la biblioteca és fàcil d'aprendre i fàcil de recordar. En la seva forma més senzilla, es poden llegir i escriure ítems OPC tan fàcilment com qualsevol variable del vostre programa Python.

- Suport a diverses plataformes, OpenOPC funciona tant amb plataformes Windows com amb plataformes no Windows. S'ha testejat amb Windows, Linux i Mac Os X.

- Té un estil de programació funcional, OpenOPC permet que les trucades OPC es puguin encadenar en un estil de programació funcional i elegant. Per exemple, es poden llegir els valors de tots els ítems que coincideixen amb un patró de comodins mitjançant una sola línia de codi Python.

- Dissenyat per a llenguatges dinàmics, la majoria d'eines d'OPC estan dissenyades per utilitzar-les amb llenguatges del sistema estàtic (com C ++ o C #), proporcionant un mapeig proper als mètodes subjacents Win32 COM. OpenOPC descarta aquest model molest i, en canvi, intenta aprofitar les funcions dinàmiques del llenguatge que proporciona Python.

Per poder instal·lar OpenOPC s'ha d'executar la següent comanda a la consola de Windows:

>> pip3 install OpenOPC — Python3x

Un cop instal·lat OpenOPC, s'hauràn instal·lat amb èxit els 4 requeriments necessaris per poder establir una connexió com a Client OPC des de Python. D'aquesta manera ja serà possible poder establir comunicació des de l'aplicació de reconeixement de veu i el PLC Omron NJ o, en aquest cas, la simulació de l'estació automatitzada amb Microsoft Excel.

3.6.5 CONFIGURACIÓ DEL CLIENT OPC AMB PYTHON

Com s'esmenta en l'apartat anterior, la configuració del Client OPC des de Python es farà amb ajuda del programa OpenOPC. Per poder utilitzar aquestes funcions, primer de tot s'ha d'establir comunicació amb el servidor KEPServerEX des de la consola d'execució de Python. Els passos a seguir per aconseguir establir aquesta connexió són els següents:

1. Primer de tot, es necessari carregar totes les llibreries necessàries per establir connexió amb l'Open OPC amb la següent comanda:
   >> import OpenOPC

2. Seguidament, s'ha de construir un objecte client OPC amb la següent comanda:
   >> opc = OpenOPC.client()

3. Un cop es crea l'objecte client OPC, es llisten tots els servidors disponibles amb la següent comanda:
   >> opc.servers()
4. Amb la llista de servidors disponibles, es connecta l'objecte client OPC amb el servidor necessari per establir la comunicació amb la següent comanda:
   >> opc.connect('Kepware.KEPServerEX.V6')

Amb aquests quatre passos descrits, s'haurà aconseguit establir una comunicació exitosa amb el servidor OPC KEPServerEX.

Il·lustració 26. Configuració del client OPC amb Python

3.6.6 CONFIGURACIÓ VARIABLES DE COMUNICACIÓ AMB PYTHON

Un cop establerta la comunicació amb el servidor OPC de KEPServerEX, es pot començar a configurar les variables de comunicació.

Per poder realitzar aquesta configuració, s'han d'executar les següents comandes:

```
>> opc.read('Nom de la variable')

>> opc.write(('Nom de la variable', valor))
```

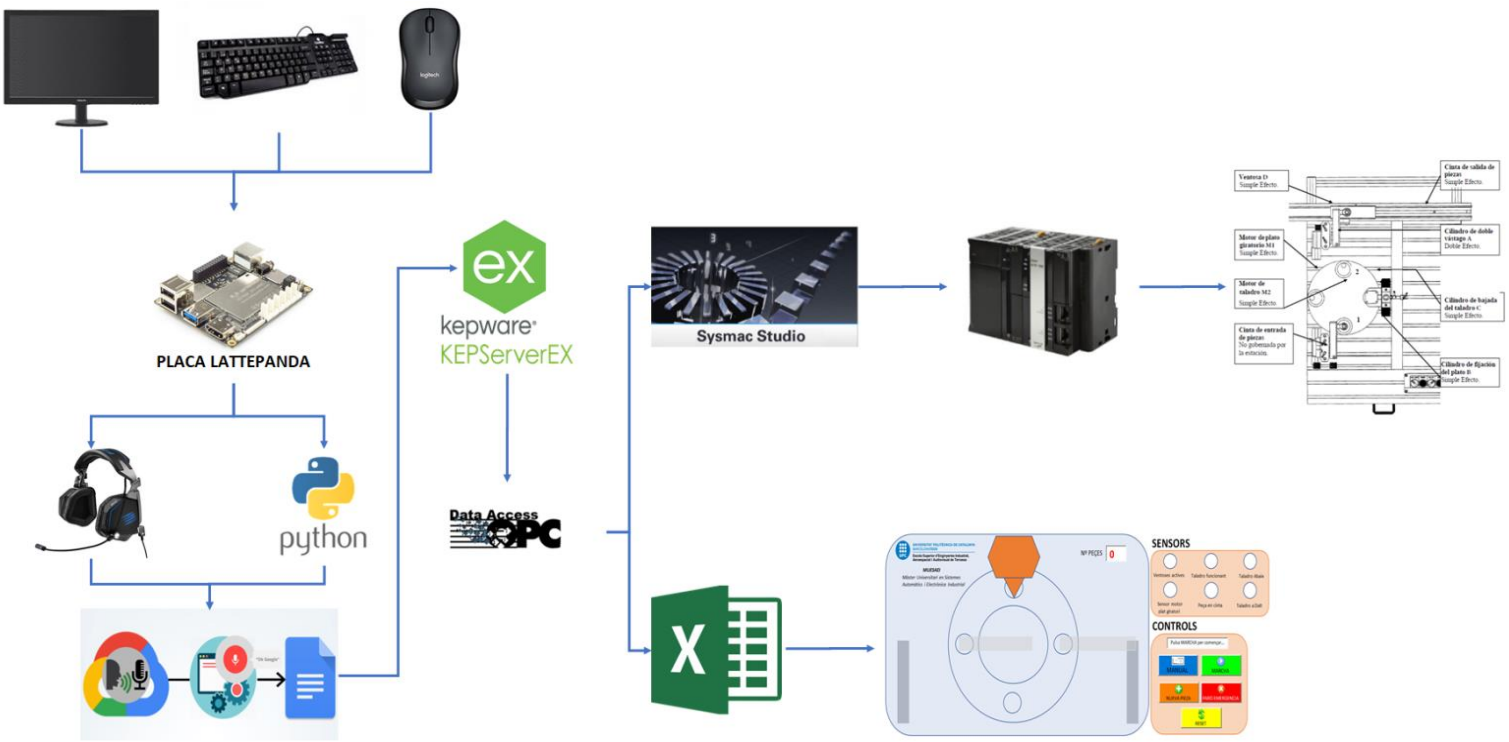
Amb la comanda "opc.read()", s'executa una lectura de la variable indicada de manera que s'obté el valor que té la variable en aquell moment. En canvi, amb la comanda "opc.write(), s'executa la escritura i, per tant el canvi de valor de la variable indicada.

4. FUNCIONAMENT GENERAL DEL SISTEMA

Degut a la pandèmia ocasionada pel COVID-19, en aquest projecte es fan servir dos mètodes similars però a la vegada diferents de connexió del servidor OPC, en aquest cas, KEPServerEX:

- Connexió servidor OPC KEPServerEX amb PLC Omron NJ
- Connexió servidor OPC KEPServerEX amb la simulació de l'estació automatitzada amb Microsoft Excel

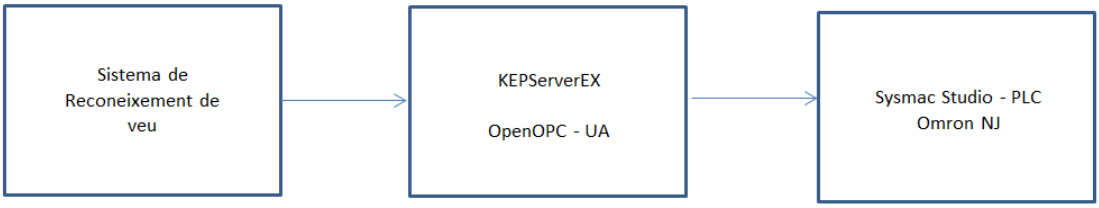
A continuació es mostrarà una imatge on es descriu tot el funcionament general del sistema, hardware i software amb les seves connexions:



Il·lustració 27. Funcionament general del sistema

4.1 CONNEXIÓ SERVIDOR OPC KEPSERVEREX AMB PLC OMRON-NJ

Gràficament es poden definir els diferents blocs d'aquest sistema de la següent manera:



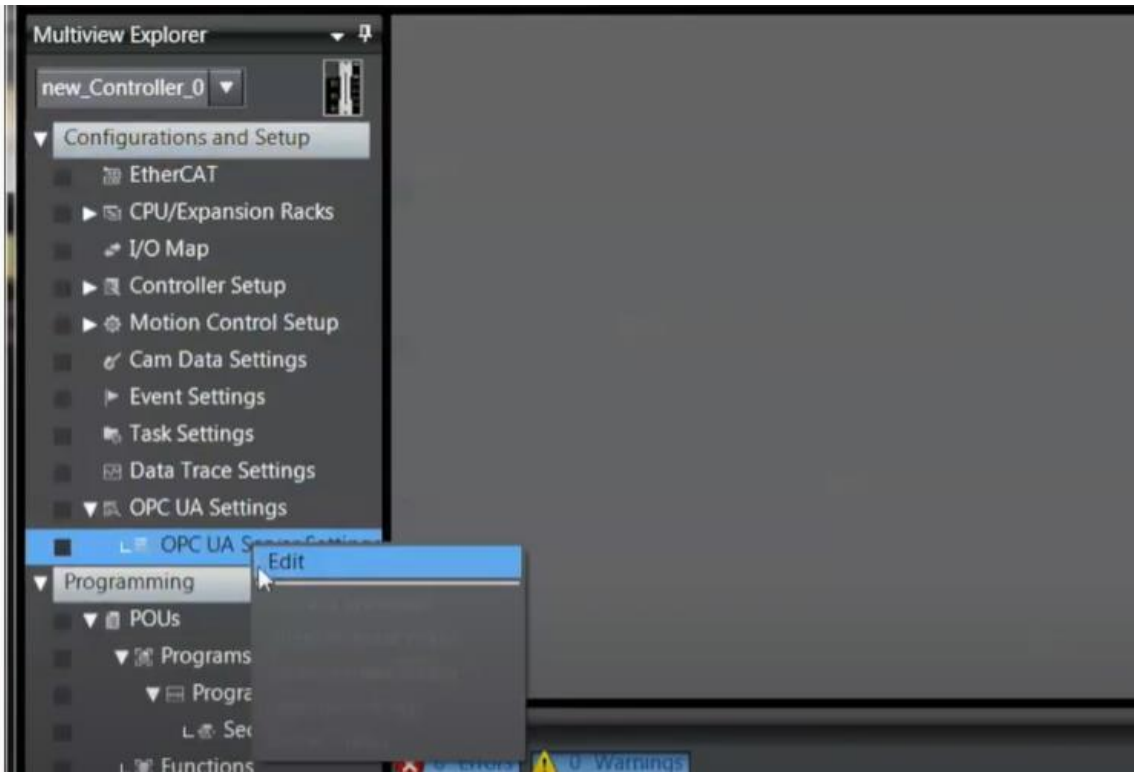
Il·lustració 28. Diagrama connexió servidor OPC KEPServerEX amb Sysmac Studio

Pel que fa la configuració del reconeixement de veu, per poder establir comunicació amb el servidor OPC KEPServerEX, com s'ha esmentat anteriorment en el punt 3.5 del projecte cal realitzar tots els passos descrits, i finalment amb dos comandes es poden llegir i escriure els valors de les variables indicades.

Un cop s'ha aconseguit establir la comunicació des de Python a KEPServerEX, s'ha d'aconseguir establir comunicació del KEPServer amb el software Sysmac Studio, el qual estarà connectat al PLC, de manera que amb el reconeixement de veu, es pugin veure els canvis realitzats a l'estació automatitzada del laboratori mitjançant el servidor OPC KEPServerEX.

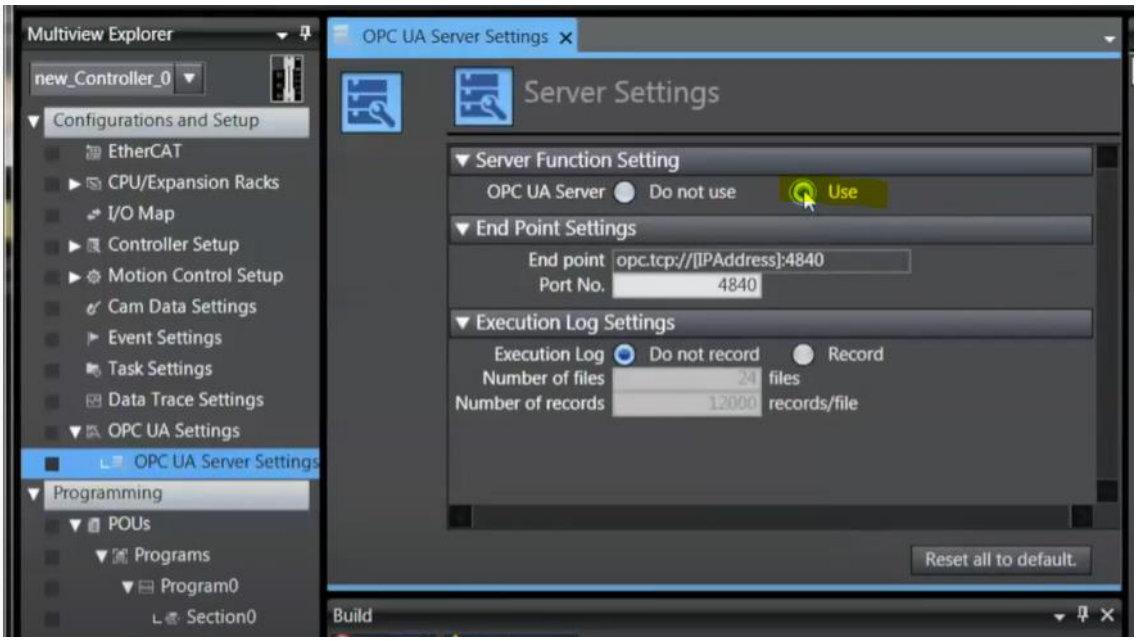
Per aconseguir establir la connexió entre el servidor OPC KEPServerEX i el PLC a través del software Sysmac Studio, primer s'ha d'activar el servidor OPC. Un cop activat el servidor OPC, en el software Sysmac Studio s'han de realitzar els següents passos:

- A "Configuracions i Ajustos" seleccionar "Ajustos OPC UA".
- Fer click al botó dret a "Ajustos OPC UA" i seleccionar "Editar".



Il·lustració 29. Configuració Servidor OPC UA Sysmac Studio

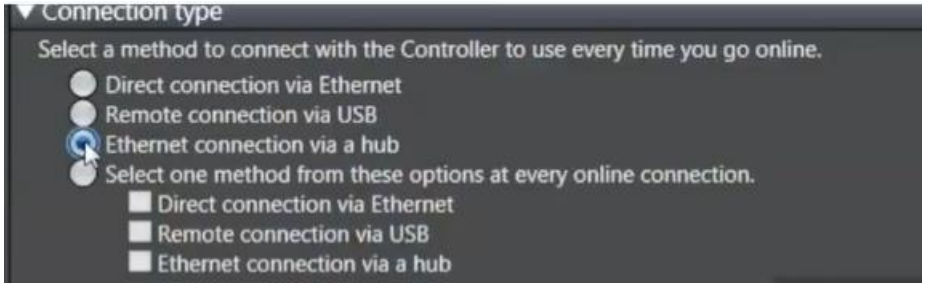
- A "Ajustos del Servidor" seleccionar "Utilitzar"



Il·lustració 30. Ajustos del servidor OPC UA Sysmac Studio

Una vegada es tingui el projecte definit i compilat, s'ha de connectar al controlador per configurar la comunicació. Els passos següents a seguir són:

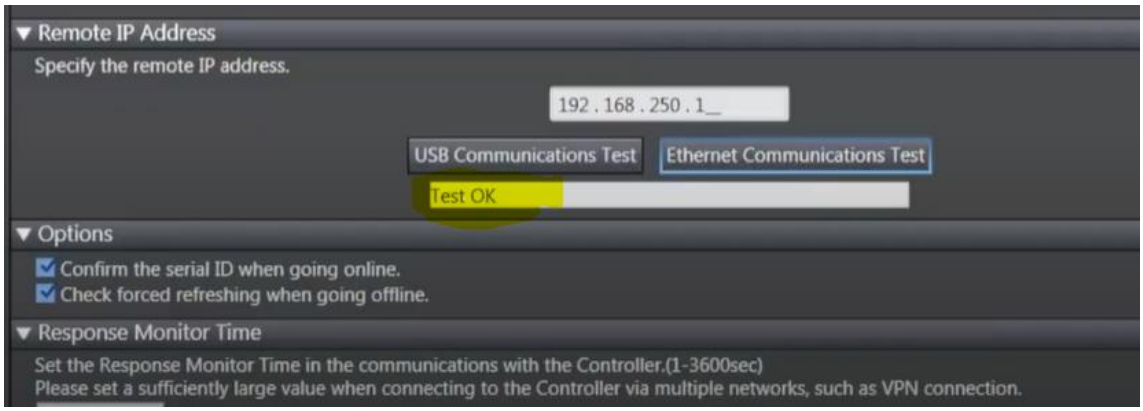
- A l'apartat "Tipus de connexió", seleccionar "Connexió Ethernet a través d'un Hub"



Il·lustració 31. Apartat "Tipus de connexió" Sysmac Studio

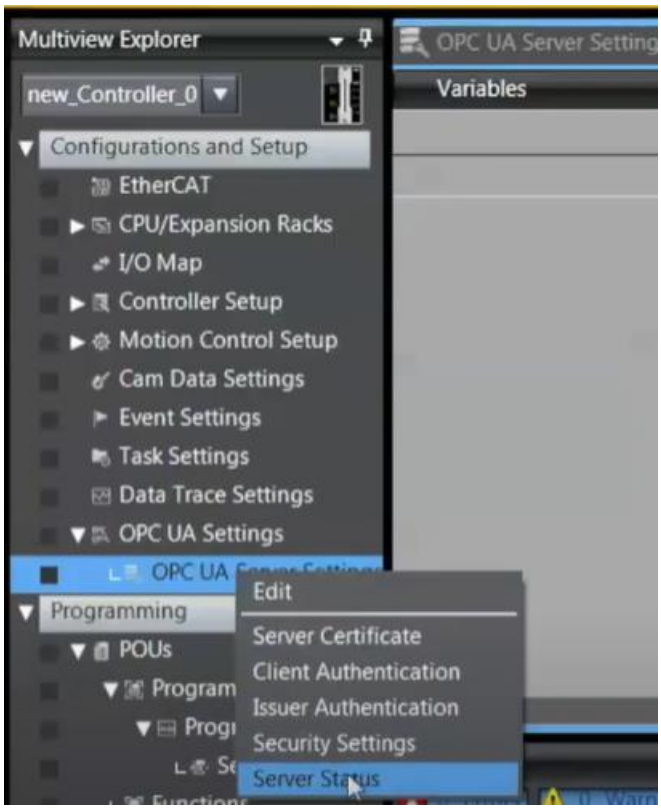
- Introduir IP del controlador a l'apartat "Adreça IP remota", on es podrà fer un test de connectivitat per comprovar que tot està correctament.

- Fer comprovació seleccionant la opció "Prova de comunicació Ethernet". En cas que tot estigui correcte, apareixerà "Prova OK" al quadre de text.



Il·lustració 32. Prova de comunicació Ethernet Sysmac Studio

Un cop s'hagi connectat el projecte amb el controlador OPC, aquest agafarà el nom original i es podrà mirar l'estat del servidor fent click amb el botó dret a l'opció "Ajustos servidor OPC UA" i seleccionar l'opció "Estat del servidor".

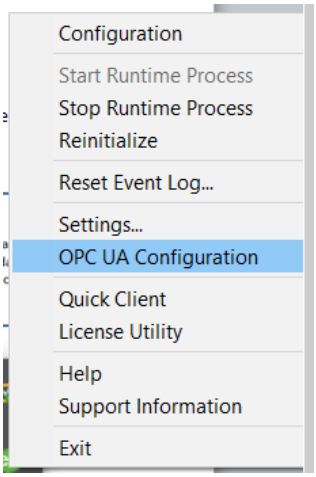


Il·lustració 33. Estat del servidor OPC UA després d'establir connexió amb el controlador Sysmac Studio

A l'estat del servidor OPC UA, es pot observar com es veu que està en funcionament i que el servidor OPC UA està en ús.

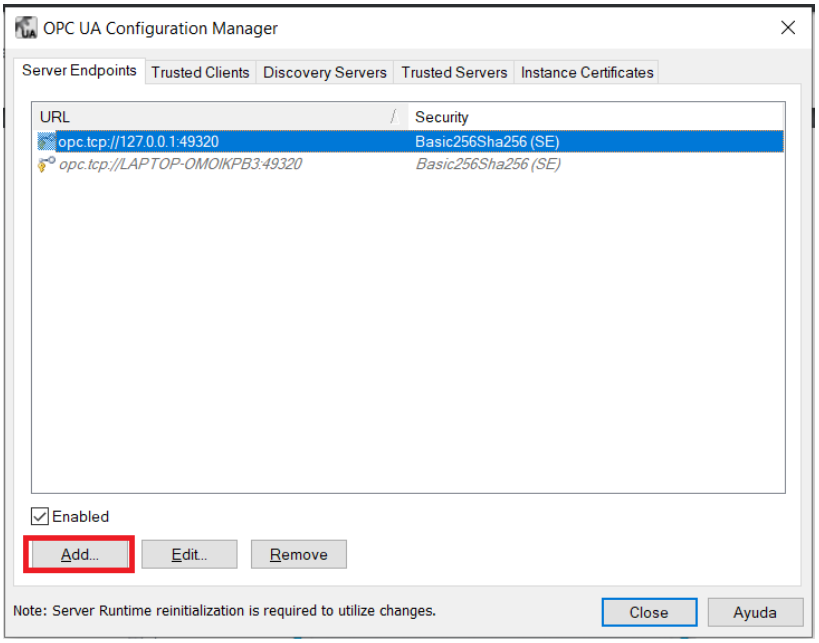
Abans de configurar el projecte de KEPServerEX, s'ha de crear el "endpoint" al servidor. Aquest "endpoint" ha de coincidir amb el que es defineixi en el controlador OPC UA client. Per a la seva creació (Carrill, 2018):

- Fer click amb el botó dret a la icona de "KEPServerEX Administration" i seleccionar "OPC UA Configuration..."



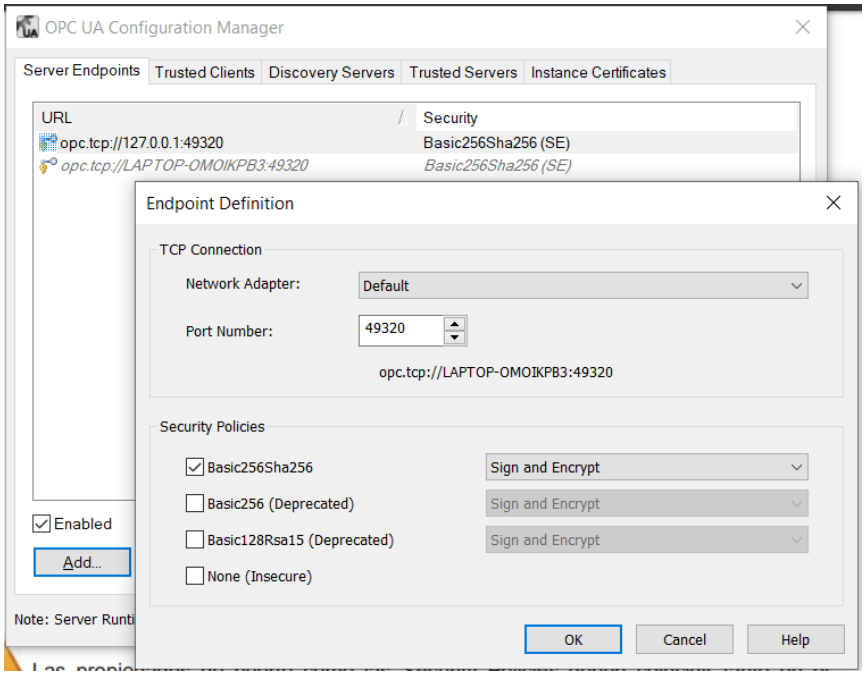
Il·lustració 34. Configuració OPC UA KEPServerEX (Carrill, 2018)

- A la pestanya “Server Endpoints, seleccionar “Add”.



Il·lustració 35. OPC UA Configuration Manager KEPServerEX (Carrill, 2018)

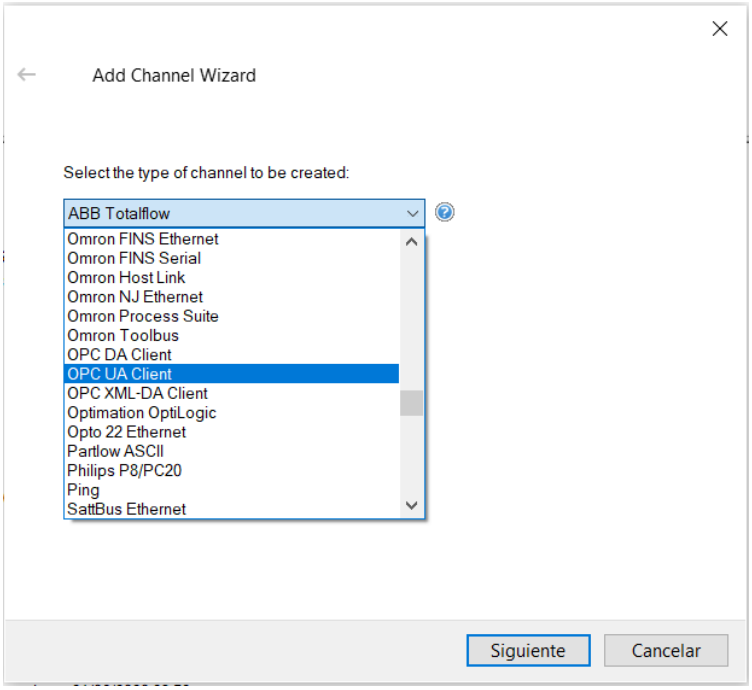
- Seleccionar la targeta de xarxa adequada
- El port 49320 està per defecte, però es pot modificar
- Seleccionar “None” i desmarcar la resta d’opcions
- Per guardar la configuració, parar el “Runtime” i iniciar-lo



Il·lustració 36. Definició de l’”endpoint” KEPServerEX (Carrill, 2018)

És important que les propietats de port com les "Security Policies" coincideixin tant en l'"endpoint" del servidor com en l'"endpoint" definit al client. Un cop s'ha creat l'"endpoint, s'ha de crear el projecte en KEPServerEX (Carrill, 2018):

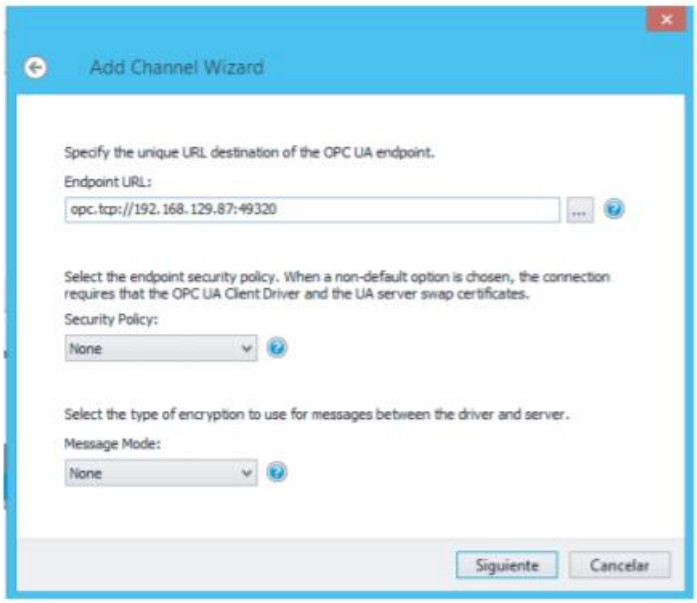
- Afegir un nou canal i seleccionar el driver OPC UA client



Il·lustració 37. Configuració "Channel" OPC UA Client KEPServerEX (Carrill, 2018)

- Deixar tots els paràmetres per defecte de "Write Optimizations", "Non-Normalized Float Handling", "UA Session" i "Authentication".

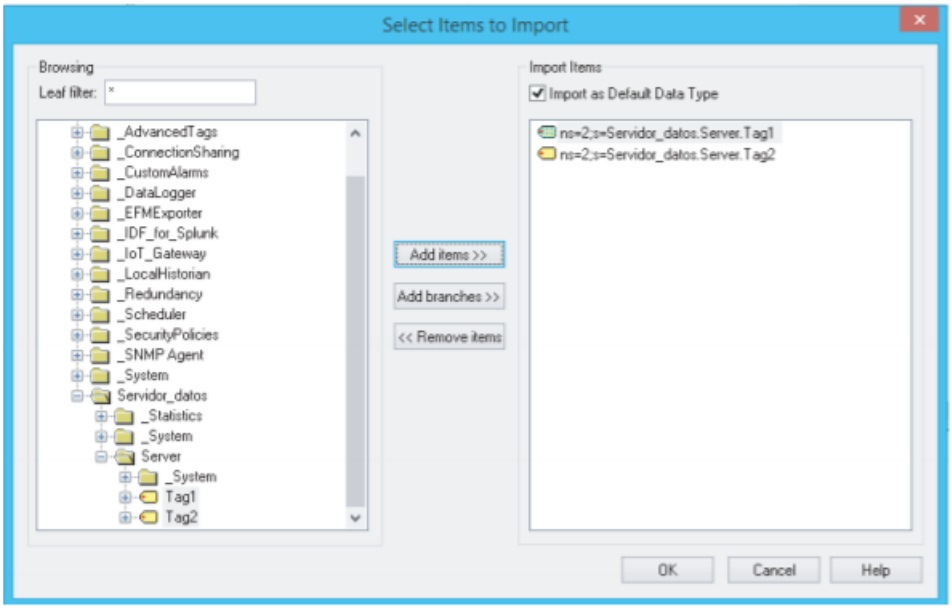
- A la pestanya de "Endpoint URL", seleccionar l'"endpoint" del servidor OPC UA i seleccionar "None" a "Security Policy".



Il·lustració 38. Configuració Endpoint URL KEPServerEX (Carrill, 2018)

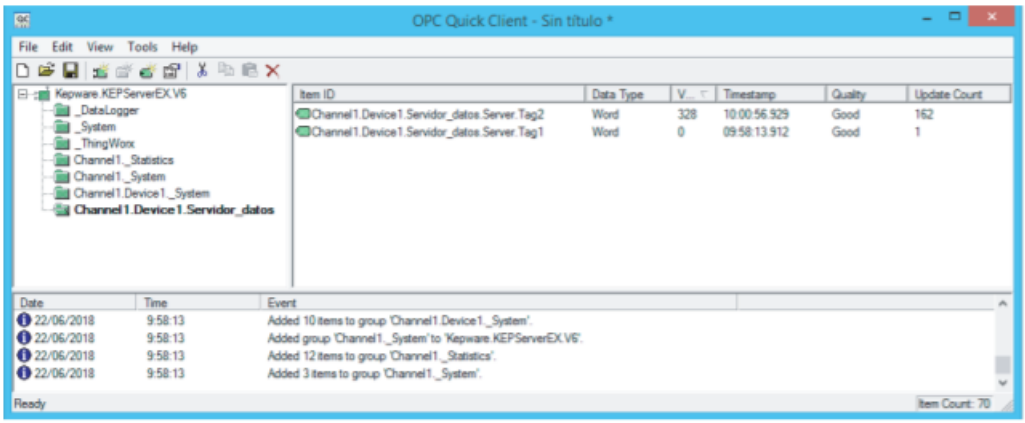
Es recomanable configurar l'"endpoint" de forma manual ja que per habilitar el cercador de servidors UA comporta més temps, ja que s'ha de habilitar el port 4840
al "Firewall". A més, no és possible realitzar el "Browse" a través de VPNs (Carrill, 2018).

- Afegir un dispositiu i deixar tots els paràmetres per defecte
- Seleccionar "Select import items...". Si s'ha configurat correctament, ha d'aparèixer la finestra amb els "tags" del servidor UA.
- Seleccionar els ítems i fer click a "Add items".



Il·lustració 39. Configuració items del dispositiu OPC UA KEPServerEX

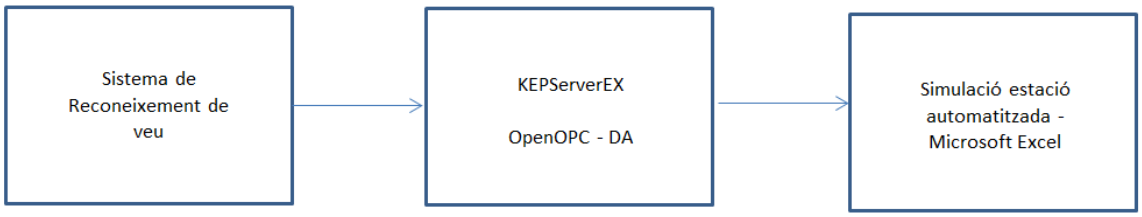
- Per comprovar que hi ha una bona connexió, obrir l'OPC Quick Client i verificar que la connexió té èxit llegint els valors de les etiquetes de KEPServerEX a l'aplicació de Client OPC UA



Il·lustració 40. Comprovació connexió a través de OPC Quick Client

4.2 CONNEXIÓ SERVIDOR OPC KEPSERVEREX AMB MICROSOFT EXCEL

En aquest cas, també es poden definir gràficament els diferents blocs de la següent manera:



Il·lustració 41. Diagrama connexió servidor OPC KEPServerEX amb Microsoft Excel

Pel que fa la configuració del reconeixement de veu, per poder establir comunicació amb el servidor OPC KEPServerEX, s'ha esmentat anteriorment en el punt 3.5 del projecte cal realitzar tots els passos descrits, i finalment amb dos comandes es poden llegir i escriure els valors de les variables indicades.

Un cop s'ha aconseguit establir la comunicació des de Python a KEPServerEX, s'ha d'aconseguir establir comunicació del KEPServer amb la simulació de l'estació automatitzada amb Microsoft Excel de manera que amb el reconeixement de veu, es pugin veure els canvis realitzats a la simulació de l'estació automatitzada mitjançant el servidor OPC KEPServerEX.

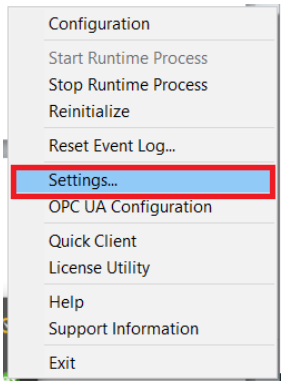
Els requisits per poder realitzar connexions DDE al servidor OPC són els següents (Kepware Developers, 2019):

- Al quadre de diàleg “Característiques d’instal·lació” de la instal·lació del servidor, s’ha de sel·leccionar DDE (Dynamic Data Exchange) a l’arbre d’interfícies del client natives.
- S'ha d'utilitzar un client DDE vàlid (com qualsevol versió d'Excel).
- És molt important executar l’arxiu d’Excel com a Administrador per permetre l’accés DDE al servidor DDE KEPServerEX.

4.2.1 CONFIGURACIÓ DEL SERVIDOR PER A LA CONNECTIVITAT DDE

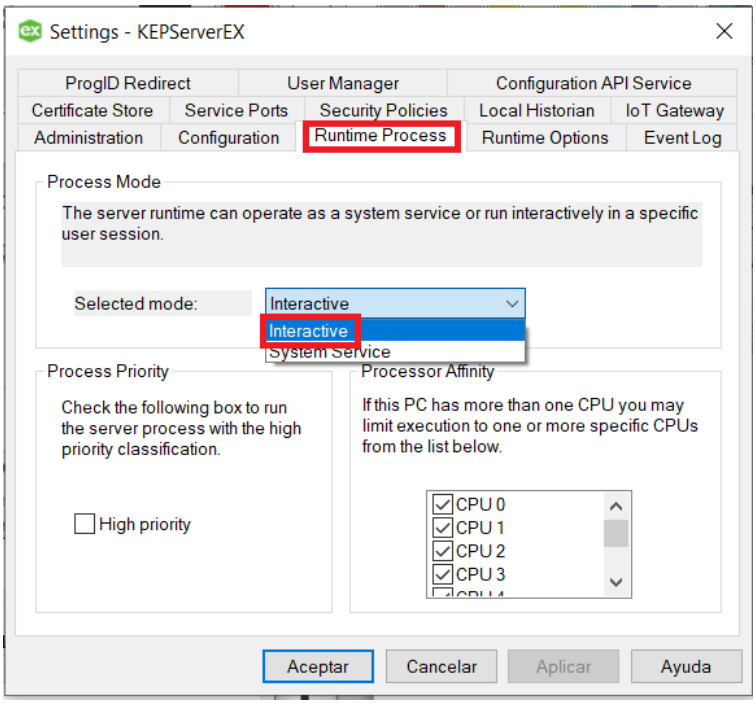
Tot i que el mode de procés per defecte del servidor és el servei del sistema, les comunicacions entre Excel i el servidor són més senzilles en mode interactiu. Per canviar el mode de procés al mode interactiu, s’han de realitzar els següents passos (Kepware Developers, 2019):

1. Fer clic amb el botó dret al menú Administració i seleccionar “Configuració...”



Il·lustració 42. Menú d’Administració de KEPServerEX

2. Obrir la pestanya Procés d’execució i localitzar l’àrea del mode de procés.
3. Utilitzar el menú desplegable “Mode seleccionat” per seleccionar Interactiu.
4. Fer clic a “D'acord”.

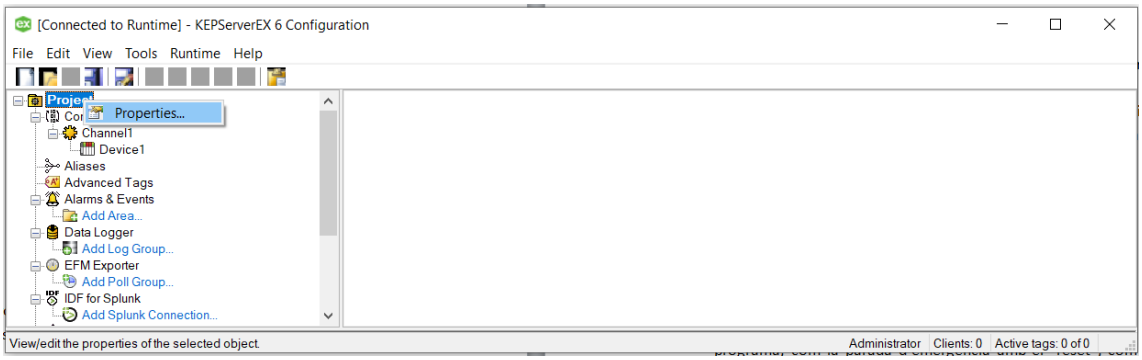


Il·lustració 43. Procés de canvi a Mode Interactiu de KEPServerEX

4.2.2 CONFIGURACIÓ DE LES PROPIETATS DDE

La instal·lació del servidor desactiva DDE de manera predeterminada perquè pot retardar el rendiment del servidor i provocar problemes de seguretat quan està habilitat innecessàriament. És necessari per a la connexió a una base de dades, seguir aquests passos per habilitar DDE:

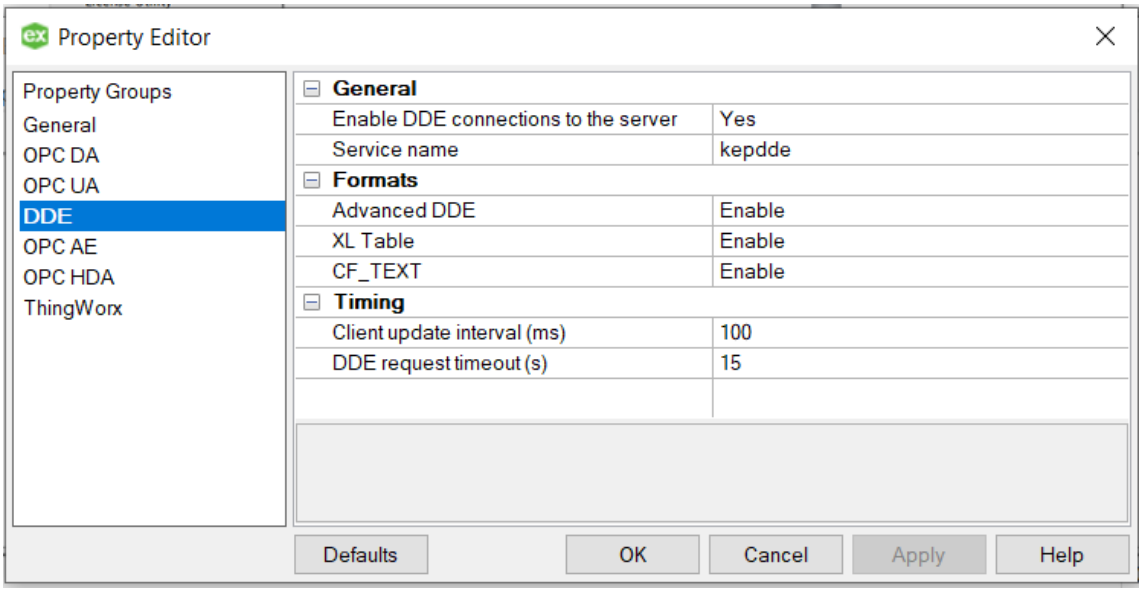
1. Triar Fitxer | Fer clic amb el botó dret a la carpeta del projecte.



Il·lustració 44. Interfície Projecte KEPServerEX

2. Seleccionar Propietats.
3. Seleccionar la pestanya DDE.
4. A General, seleccionar Sí per habilitar les connexions DDE al servidor.
5. A Nom de servei, guardar el nom predeterminat "kepdde".
6. A l'àrea de Formats, habilitar les tres opcions disponibles abans de continuar.

7. Deixar les propietats restants a la configuració predeterminada.
8. Fer clic a “D'acord”.
9. Reiniciar el servidor perquè els canvis siguin efectius.



Il·lustració 45. Procés de configuració de les propietats DDE

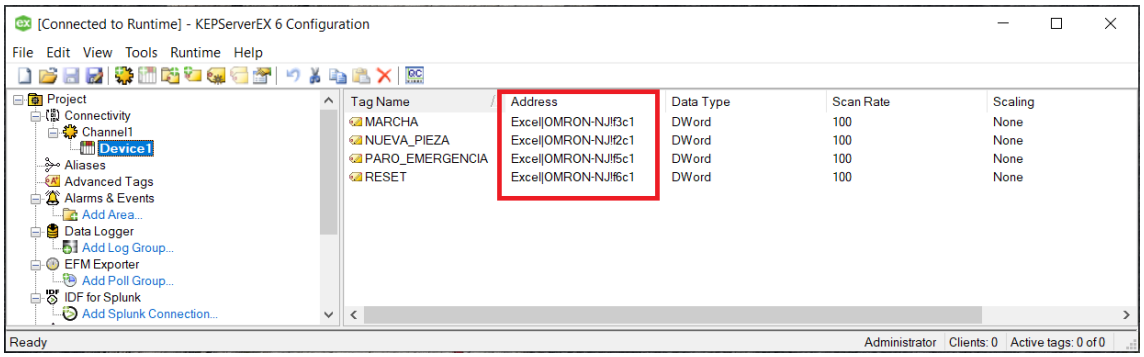
4.2.3 ESCRIPTURA I ACCÉS A DADES DE DDE EN EXCEL

Una connexió DDE a Excel requereix un nom, un tema i un element de l'aplicació. Tanmateix, només cal un nom de tema a KEPServerEX: es pot accedir a totes les dades del servidor mitjançant "_ddedata" com a tema de DDE. Aquest tema global del servidor permet accedir a totes les etiquetes o adreces del servidor, independentment del dispositiu o grup d'etiquetes on es creï.
Amb aquest tema en un enllaç, s’ha de proporcionar un identificador de ruta complet per a l’element previst. L’identificador de ruta ha d’incloure grups d’etiquetes, subgrups, canals i noms de dispositiu (Kepware Developers, 2019).

És, per tant, que només afegint a l’adreça dels actuadors o variables del sistema en aquest cas, el nom del software en qüestió, el nom de la pàgina i la cel·la de la variable, ja estarà establerta la connexió amb el software Microsoft Excel. Els passos a realitzar per poder establir aquesta connexió són els següents:

1. Fer clic amb el botó dret a sobre del actuador o variable del sistema.
2. Seleccionar “Propietats”.
3. Seleccionar “Data properties”.
4. Establir una adreça amb la següent estructura:

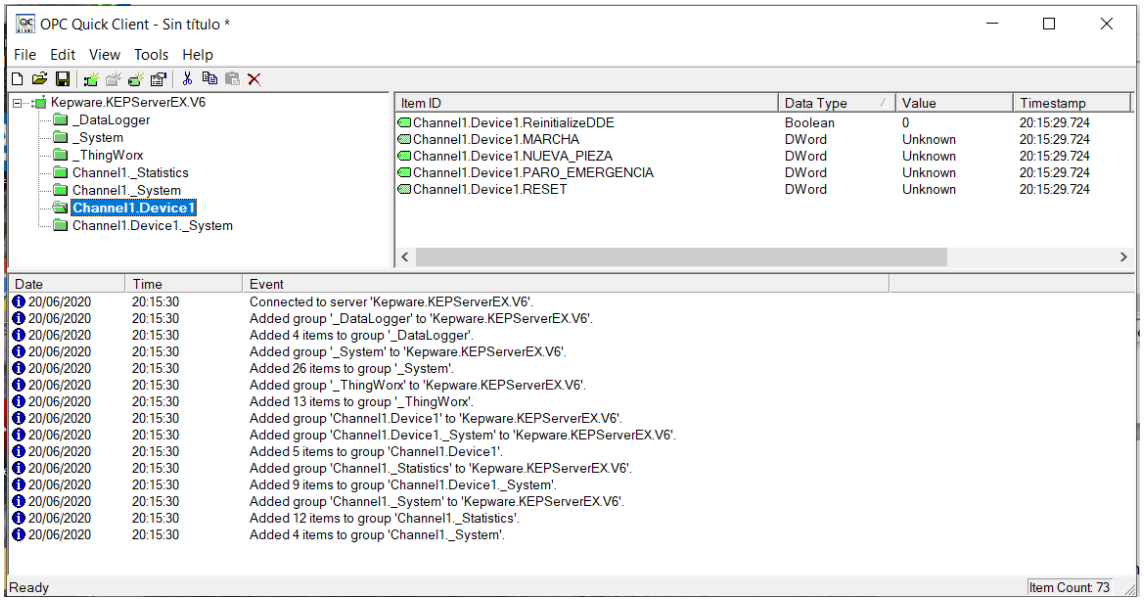
   Nom del Servei DDE|Nom del tema!Nom de l’element



Il·lustració 46. Exemple de connectivitat KEPServerEX amb Microsoft Excel

Un cop establert i acabat el procés d'escriptura i accés a dades DDE en Excel, s'haurà d'accedir a OPC Quick Client, ja que serà mitjançant OPC Quick Client per on es podrà interactuar amb les variables de la Simulació de l'estació Automatitzada. Els passos per accedir a OPC Quick Client, i a les variables del sistema són els següents:

1. Obrir OPC Quick Client
2. Seleccionar "Channel1.Device1



Il·lustració 47. Variables del sistema a OPC Quick Client

Un cop obert OPC Quick Client, a la columna "Value", es podrà veure i modificar tant manualment, com a través del sistema de reconeixement de veu el valor de les quatre variables del sistema.

4.3 FUNCIONAMENT DE L'APLICACIÓ AMB PYTHON

El programa realitzat amb Python el qual l'objectiu es transcriure la veu a text a partir d'un arxiu d'àudio, ha sigut creat a partir de varies funcionalitats diferents. Les diferents funcions que formen el programa de reconeixement de veu són:

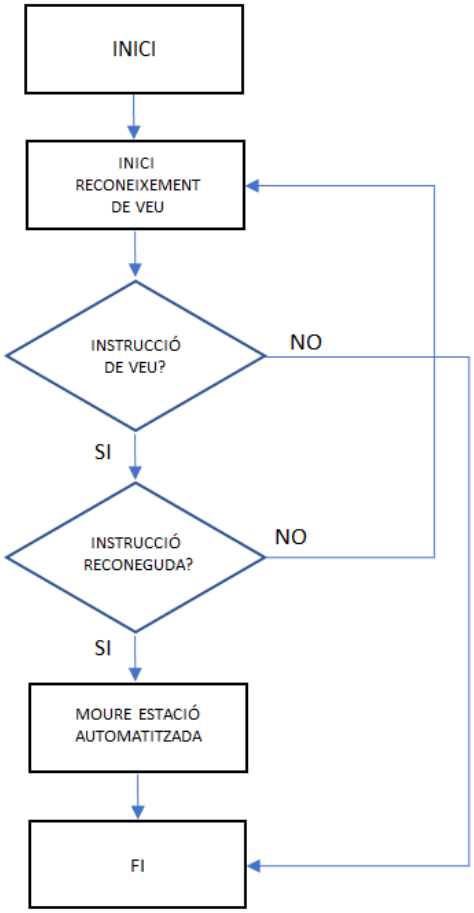
- Entrenament del reconeixement de veu
- Establir comunicació amb el servidor OPC de KEPServer
- Reconeixement de veu
- Utilització de l'API de Google Cloud
- Transcripció de l'arxiu de reconeixement de veu
- Identificació d'instruccions

Amb totes aquestes funcions s'ha aconseguit generar un programa funcional que aconsegueix poder transcriure pràcticament tot el que escolta fins que l'aplicació no rep cap paraula en un període curt de temps.

A continuació es mostra la seqüència lògica que s'ha seguit per arribar a l'objectiu d'aquesta aplicació:

1. Inicia l'aplicació iniciant el sistema de Control a través del reconeixement de veu.
2. S'estableix la connexió amb el servidor OPC.
3. S'estableix la comunicació amb Google Cloud.
4. L'aplicació demana una instrucció la qual es començarà a gravar pel micròfon. Aquesta instrucció hauria de ser preferiblement "Marcha", si no el procés seqüencial no s'iniciarà.
5. A partir de la instrucció demanada, es crearà un arxiu d'àudio, el qual serà enviat al servidor de Google Cloud i s'obtindrà una transcripció de la instrucció realitzada.
6. Si aquesta transcripció coincideix amb alguna de les instruccions possibles, per exemple, Marcha, canviarà el valor de la variable Marcha de 0 a 1, i s'iniciarà el procés del sistema.
7. En tot moment l'operari pot realitzar la instrucció "Paro de Emergencia" per parar completament el procés o bé la instrucció "Terminar" per acabar la sèrie de peces pendents (canviant el valor de la variable Marcha de 1 a 0), així com la instrucció "Salir" per sortir de l'aplicació de reconeixement de veu.

A continuació es mostra un diagrama de flux sobre la seqüència lògica del funcionament de l'aplicació amb Python:



Il·lustració 48. Diagrama de flux del funcionament de l'aplicació de reconeixement de veu

4.4 PROGRAMACIÓ PLC OMRON-NJ

L'objectiu d'aquesta pràctica és realitzar l'automatització d'aquesta estació de manera que quan es premi el polsador de "marxa" de l'estació es realitzi el funcionament de la mateixa depenent de la posició de l'interruptor "man/auto" (realització de la seqüència pas a pas segons premem "marxa" o de manera contínua, respectivament). La seqüència ha de parar-se immediatament quan es produeix una "parada d'emergència" i tornar a condicions inicials quan es produeix un "reset"

El propòsit del programa que executa el PLC Omron-NJ, es poder controlar-lo mitjançant l'aplicació del reconeixement de veu.

Com s'ha esmentat anteriorment, la estació II és l'encarregada de foradar les peces subministrades per l'estació I, per això disposa d'un cilindre de doble tija que és l'encarregat de recollir i expulsar les peces de l'estació de manera simultània. Aquestes peces es depositen sobre un plat giratori que les

posiciona a sota d'un taladro que serà l'encarregat de foradar les peces, les quals han sigut prèviament mitjançant un cilindre de simple efecte.

La seqüència a realitzar serà la següent:

1. El cilindre de doble tija (A), que és un cilindre giratori, es dirigeix a recollir una peça de la cinta transportadora controlada per l'estació I. Al ser un cilindre de doble tija, també recull una peça del plat giratori per a ser dipositada sobre la cinta transportadora de sortida de l'estació.

2. S'activa la ventosa (D) d'aquest cilindre per recollir la peça de la cinta transportadora i del plat giratori.

3. El cilindre de doble tija (A) es dirigeix a la posició de deixar peça.

4. Es desactiva la ventosa (D) alliberant les peces, que es deixen, sobre el plat giratori i sobre la cinta transportadora.

5. S'activa el motor del plat giratori (M1) per situar la peça a tractar sota el Taladro aquest motor es manté activat fins que el sensor de posició del plat s'activa.

6. A continuació, s'activa el motor del Taladro (M2).

7. Es fixa la peça en el seu compartiment sobre el plat giratori, expulsant el cilindre de fixació de peça (B).

8. Un cop fixada la peça es baixa el Taladro activant el cilindre de baixada de Taladro (C).

9. Quan el cilindre arriba al final del seu recorregut, es puja el Taladro (C).

10. A continuació, es para el motor del Taladro (M2).

11. Es desbloqueja el cilindre (b) de fixació de peça i la estació es troba novament en condicions inicials.

A continuació, a la il·lustració 48 es mostrarà l'estat inicial del sistema:

DESCRIPCION	ESTADO INICIAL
Cilindro A (doble efecto, doble vástago)	A-, a0=1
Cilindro B (simple efecto)	B-, b0=1
Cilindro C (simple efecto)	C-, c0=1
Ventosa D (simple efecto)	D-
Motor cinta M3 (simple efecto)	Activado
Motor plato giratorio M1 (simple efecto)	Desactivado
Motor del taladro (simple efecto)	Desactivado
Sensor posición plato	Activado
Sensor de presencia de pieza en cinta	Activado
Sensor de presencia de pieza en plato	Desactivado
Válvula progresiva	Activado

Il·lustració 49. Estat inicial del sistema

El programa creat per aquest projecte consta de tres parts:

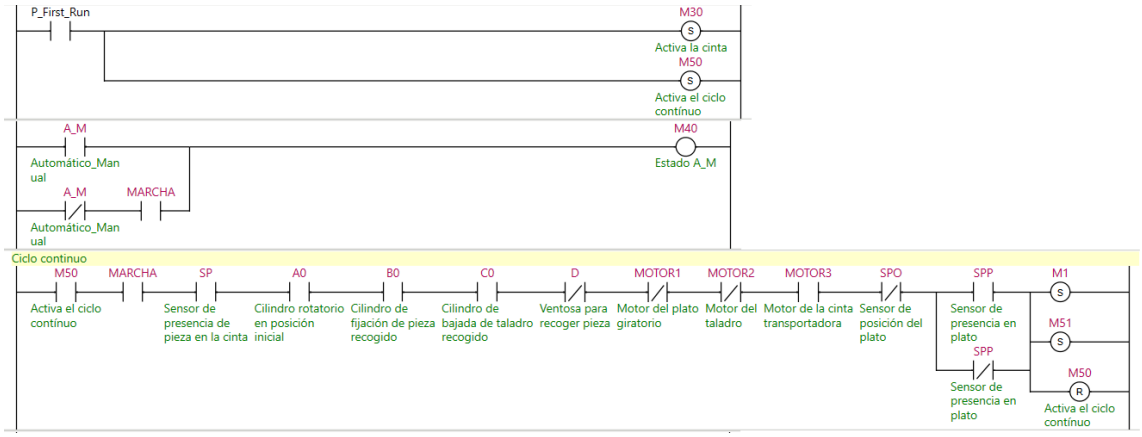
- Inicialització: En aquesta part del programa es troba tant el cicle continu del programa, com la parada d'emergència amb el "reset", com l'estat inicial del que serà el cos del programa del PLC.

- Cos de programa: En aquesta part es troba tot el cicle SCAN del programa, és a dir, tota la seqüència que realitzarà l'estació automatitzada.

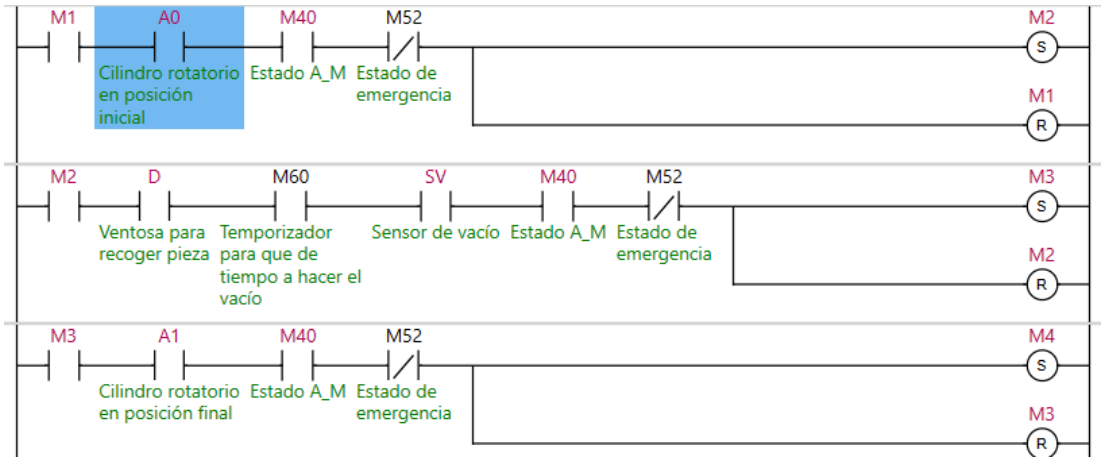
- Sortides: En aquesta última part es troben totes les bobines, les quals fan referència a les sortides físiques del PLC, tant de les variables del cos de programa com poden ser també les funcions lògiques complexes, com per exemple, els temporitzadors, comptadors, etc..

Per realitzar la programació del PLC, primerament s'ha realitzat un petit Grafcet, i a partir d'aquest, s'ha traslladat al llenguatge de programació Ladder. D'aquesta manera, el Ladder ha sigut programat per etapes, i cada etapa està representada per "MX", i la resta són les condicions de transició que s'han de complir per poder passar d'una etapa a una altra etapa.

A continuació es veurà un exemple per passar de l'etapa M1 (Inici del sistema prement el botó Marxa) a M2:



Il·lustració 50. Estat inicial del sistema a Sysmac Studio

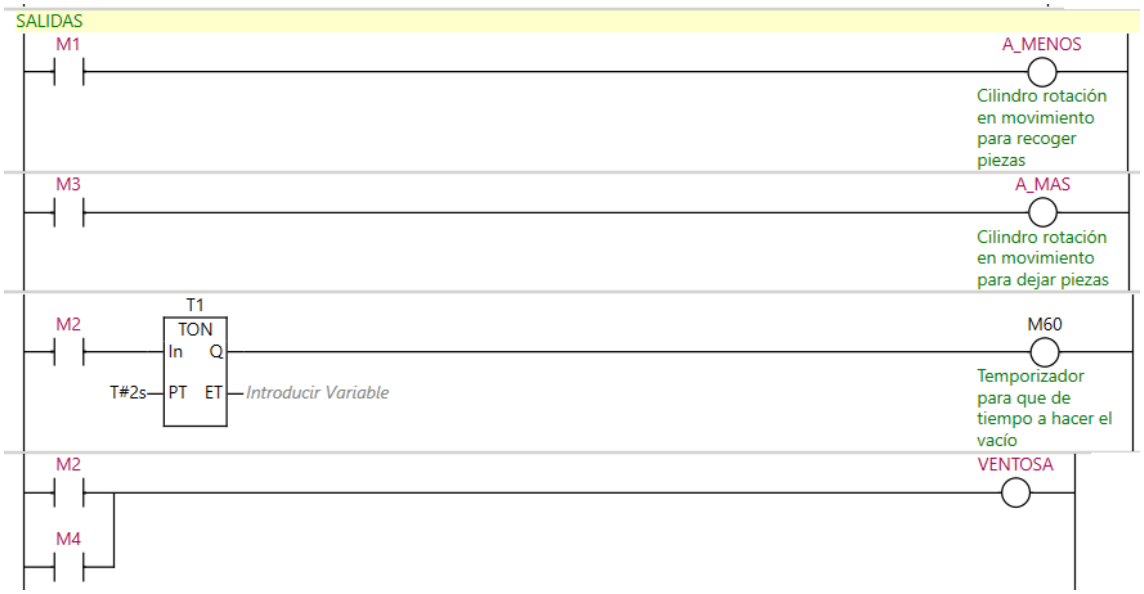


Il·lustració 51. Seqüència del cos de programa Sysmac Studio

Per exemple, per poder passar d'M1 a M2, primerament, M1 ha d'estar actiu. M1 estarà actiu si es compleixen les condicions inicials i si el botó Marxa és premut. Un cop M1 és actiu, s'han de complir les condicions A0, és a dir, el cilindre ha d'estar a la posició per recollir peça, si l'estat és automàtic o manual, i si no s'ha fet cap parada d'emergència. Un cop totes aquestes condicions estiguin actives, el sistema passarà a l'etapa M2, desactivant o fent un reset a l'etapa M1.

El programa té 12 etapes, que són els 11 passos que té la seqüència de l'estació més 1 etapa que és la de les condicions inicials.

Finalment, a les sortides es troben les etapes "MX" i també es troben els temporitzadors o els comptadors associats a cada sortida física que té el PLC, en aquest cas, l'estació automatitzada.



Il·lustració 52. Etapes associades a les sortides físiques del PLC amb Sysmac Studio

Pel que fa a les variables del sistema, es poden definir en dos tipus diferents de variables:

- Variables globals: La variables és accessible des de tots els sub-programes de l'aplicació
- Variables internes: La variable només és accessible des del sub-programa al que pertany, no pot ser llegida o modificada des d'un altre sub-programa.

En aquest programa, totes les variables són globals perquè pràcticament totes les variables estan repartides entre els tres sub-programes que componen el programa del PLC, menys els temporitzadors i el comptador, ja que, només es troben en el sub-programa "salidas".

Les variables globals del sistema són les següents:

Nombre	Tipo de datos	Constante	Comentario
A_M	BOOL	☐	Automático_Manual
A_MAS	BOOL	☐	Cilindro rotación en movimiento para dejar piezas
A_MENOS	BOOL	☐	Cilindro rotación en movimiento para recoger piezas
A0	BOOL	☐	Cilindro rotatorio en posición inicial
A1	BOOL	☐	Cilindro rotatorio en posición final
B	BOOL	☐	
B0	BOOL	☐	Cilindro de fijación de pieza recogido
B1	BOOL	☐	Cilindro de fijación de pieza extendido
C	BOOL	☐	
C0	BOOL	☐	Cilindro de bajada de taladro recogido
C1	BOOL	☐	Cilindro de bajada de taladro extendido
CINTA	BOOL	☐	
D	BOOL	☐	Ventosa para recoger pieza
E	BOOL	☐	
Emergencia	BOOL	☐	
Enfuncionamiento	BOOL	☐	
Enparo	BOOL	☐	
M1	BOOL	☐	
M10	BOOL	☐	
M11	BOOL	☐	
M12	BOOL	☐	
M2	BOOL	☐	
M3	BOOL	☐	
M30	BOOL	☐	Activa la cinta
M4	BOOL	☐	
M40	BOOL	☐	Estado A_M
M5	BOOL	☐	
M50	BOOL	☐	Activa el ciclo contínuo
M51	BOOL	☐	
M6	BOOL	☐	
M7	BOOL	☐	
M8	BOOL	☐	
M9	BOOL	☐	
MARCHA	BOOL	☐	
MOTOR_1	BOOL	☐	
MOTOR_2	BOOL	☐	
MOTOR1	BOOL	☐	Motor del plato giratorio
MOTOR2	BOOL	☐	Motor del taladro
MOTOR3	BOOL	☐	Motor de la cinta transportadora
Numeropiezas	INT	☐	Número de piezas fabricadas
RESET	BOOL	☐	
SP	BOOL	☐	Sensor de presencia de pieza en la cinta
SPO	BOOL	☐	Sensor de posición del plato
SPP	BOOL	☐	Sensor de presencia en plato
SV	BOOL	☐	Sensor de vacío
V_MAS	BOOL	☐	Aire en las válvulas
VENTOSA	BOOL	☐	

Il·lustració 53. Variables globals del sistema

Les variables internes del sistema són les següents:

valor_actual	INT			☐	☐
T2	TON			☐	☐
T1	TON			☐	☐
mas_uno	INT			☐	☐
M70	BOOL			☐	☐
M60	BOOL			☐	☐

Il·lustració 54. Variables internes o locals del sistema

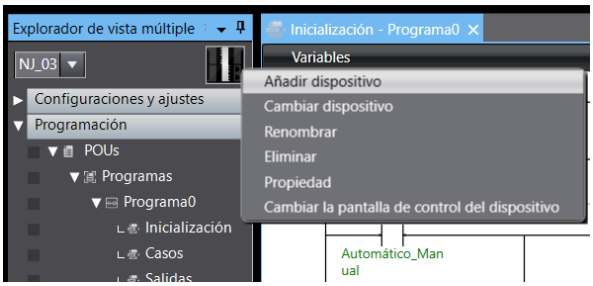
4.4.1 DESENVOLUPAMENT HMI DEL SISTEMA

Per demostrar el correcte funcionament del programa, s'ha decidit crear una pantalla HMI (Human-Machine Interface). És la interfície entre el procés i els operaris d'una fàbrica, una línia de producció, una empresa o qualsevol sistema on sigui necessària l'operació per part d'un humà. En si, és un panell d'instruments que l'operari pot manipular per controlar un procés.

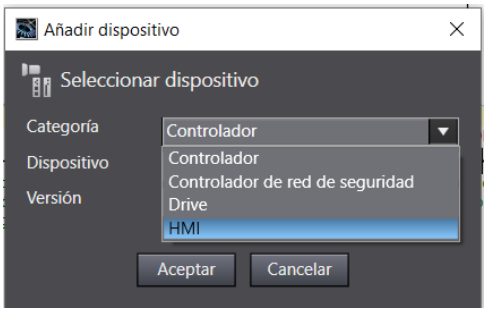
És la principal eina que utilitzen els operaris i els supervisors de línia per coordinar i controlar processos industrials i de fabricació. L'HMI tradueix variables d'un procés complex en informació útil i processable.

La funció principal dels HMI és mostrar informació en temps real, proporcionar gràfics visuals i digeribles que aportin significat i context sobre l'estat del motor, la vàlvula, nivells i altres paràmetres d'un determinat procés.

El disseny d'HMI s'ha creat també amb el software Sysmac Studio. Per poder realitzar això, primerament s'ha de crear un nou dispositiu i seleccionar HMI:

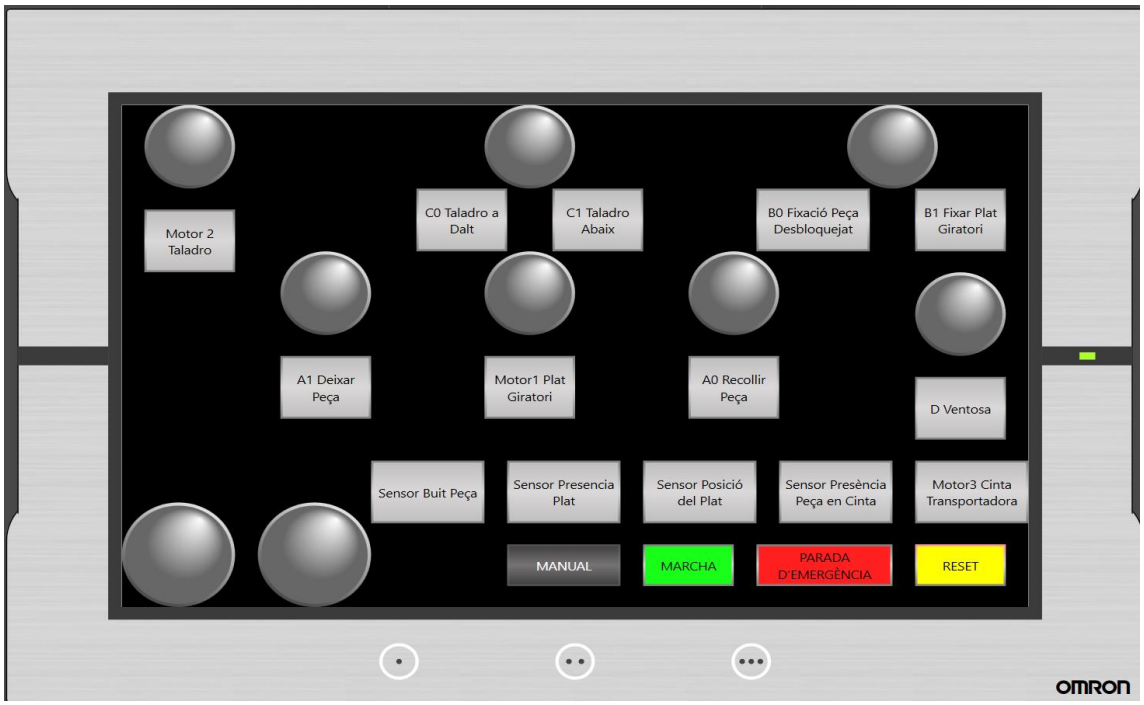


Il·lustració 55. Afegir dispositiu HMI



Il·lustració 56. Configuració dispositiu HMI

Un cop creat el dispositiu HMI, el que s'haurà de fer es crear botons, indicadors lluminosos, etc.. per poder dissenyar un HMI que demostri el correcte funcionament del programa. En aquest cas, el disseny creat del dispositiu és el següent:



Il·lustració 57. Pantalla HMI del sistema amb Sysmac Studio

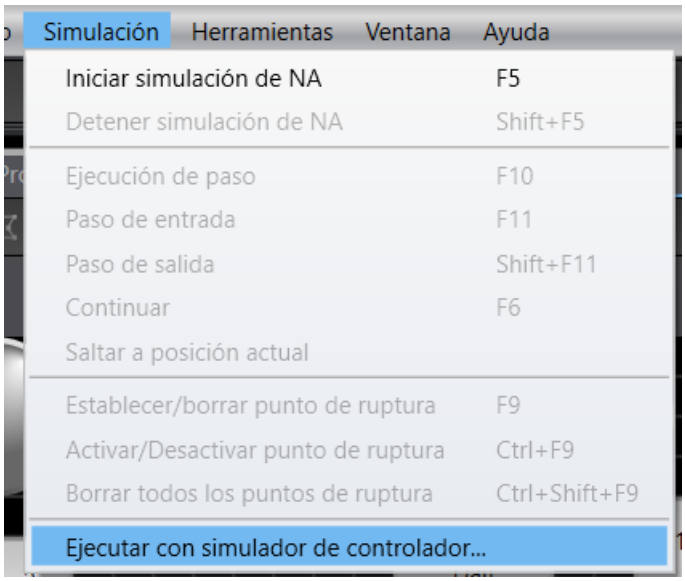
En aquest dispositiu HMI, s'han creat noves variables globals, les quals tant botons com indicadors lluminosos s'han associat a les variables globals del PLC esmentades anteriorment. Cada botó està associat a una entrada, i cada indicador lluminós està associat a una sortida. A continuació, es mostrarà la taula de variables globals associades a les variables del PLC del dispositiu HMI:

MarchaButton	Boolean		NJ_03.MARCHA	☐	☐	500 milisegundos	Ninguno
CilindroA0RecogerPieza	Boolean		NJ_03.A0	☐	☐	500 milisegundos	Ninguno
CilindroA1DejarPieza	Boolean		NJ_03.A1	☐	☐	500 milisegundos	Ninguno
CilindroB0PiezaNoFijada	Boolean		NJ_03.B0	☐	☐	500 milisegundos	Ninguno
CilindroB1FijarPieza	Boolean		NJ_03.B1	☐	☐	500 milisegundos	Ninguno
CilindroC0TaladroArriba	Boolean		NJ_03.C0	☐	☐	500 milisegundos	Ninguno
CilindroC1BajarTaladro	Boolean		NJ_03.C1	☐	☐	500 milisegundos	Ninguno
CilindroDVentosa	Boolean		NJ_03.D	☐	☐	500 milisegundos	Ninguno
SensorPresenciaPiezaCi...	Boolean		NJ_03.SP	☐	☐	500 milisegundos	Ninguno
Motor1PlatoGiratorio	Boolean		NJ_03.MOTOR1	☐	☐	500 milisegundos	Ninguno
Motor2Taladro	Boolean		NJ_03.MOTOR2	☐	☐	500 milisegundos	Ninguno
Motor3CintaTransporta...	Boolean		NJ_03.MOTOR3	☐	☐	500 milisegundos	Ninguno
SensorPosicionPlato	Boolean		NJ_03.SPO	☐	☐	500 milisegundos	Ninguno
SensorPresenciaPlato	Boolean		NJ_03.SPP	☐	☐	500 milisegundos	Ninguno
SensorVacioPieza	Boolean		NJ_03.SV	☐	☐	500 milisegundos	Ninguno
ParoEmergencia	Boolean		NJ_03.Emergencia	☐	☐	500 milisegundos	Ninguno
Reset	Boolean		NJ_03.RESET	☐	☐	500 milisegundos	Ninguno
Ventosa	Boolean		NJ_03.VENTOSA	☐	☐	500 milisegundos	Ninguno
Motor2LED	Boolean		NJ_03.MOTOR_2	☐	☐	500 milisegundos	Ninguno
CilindroB	Boolean		NJ_03.B	☐	☐	500 milisegundos	Ninguno
CilindroC	Boolean		NJ_03.C	☐	☐	500 milisegundos	Ninguno
EnFuncionamiento	Boolean		NJ_03.Enfuncionamiento	☐	☐	500 milisegundos	Ninguno
EnParo	Boolean		NJ_03.Enparo	☐	☐	500 milisegundos	Ninguno
Automatico_Manual	Boolean		NJ_03.A_M	☐	☐	500 milisegundos	Ninguno
RecogiendoPieza	Boolean		NJ_03.A_MENOS	☐	☐	500 milisegundos	Ninguno
DejandoPieza	Boolean		NJ_03.A_MAS	☐	☐	500 milisegundos	Ninguno
Motor1Led	Boolean		NJ_03.MOTOR_1	☐	☐	500 milisegundos	Ninguno

Il·lustració 58. Variables globals del dispositiu HMI

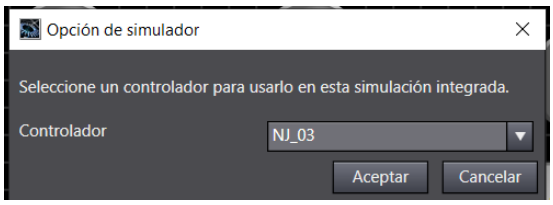
Per poder executar la simulació amb Sysmac Studio, s'han de seguir els següents passos:

- Seleccionar Simulació.
- Seleccionar Executar amb Simulació de controlador.



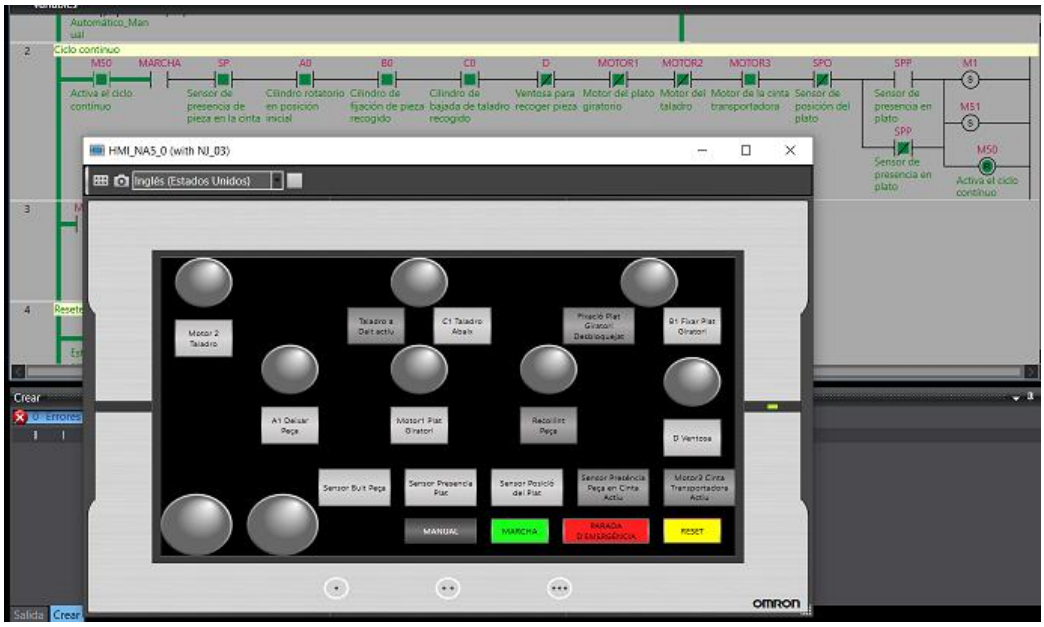
Il·lustració 59. Configuració per executar Simulació amb Sysmac Studio

- Seleccionar el controlador indicat, en aquest cas, NJ.



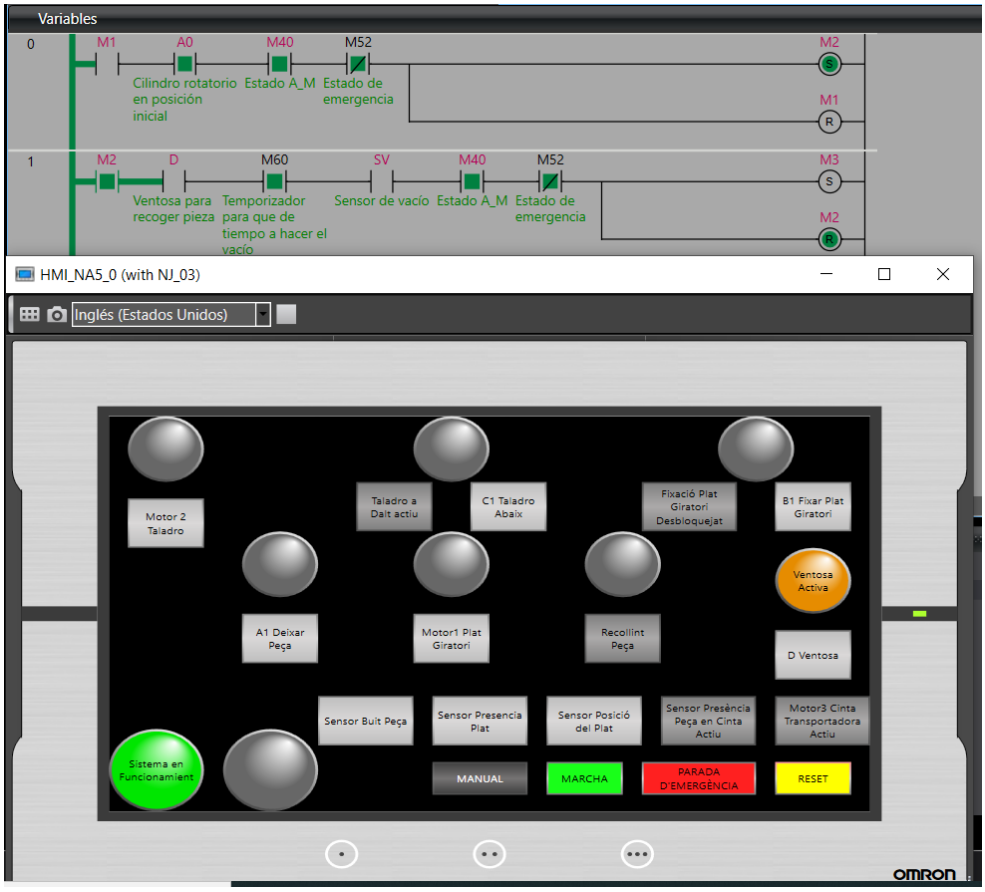
Il·lustració 60. Selecció del controlador per executar la Simulació

Un cop la simulació és executada, es pot veure el seu funcionament a través de l'HMI i del llenguatge de programació Ladder. A continuació es mostraran dos exemples de com es pot demostrar el correcte funcionament del programa mitjançant l'HMI i Ladder:



Il·lustració 61. Condicions inicials del programa Ladder/HMI

Com es pot observar, totes les condicions inicials de l'HMI estan actives, i també ho estan al Ladder. En aquest moment el sistema està esperant que el botó Marxa sigui premut per poder començar la seqüència. A continuació es mostra una imatge del moment posterior a prémer el botó Marcha:

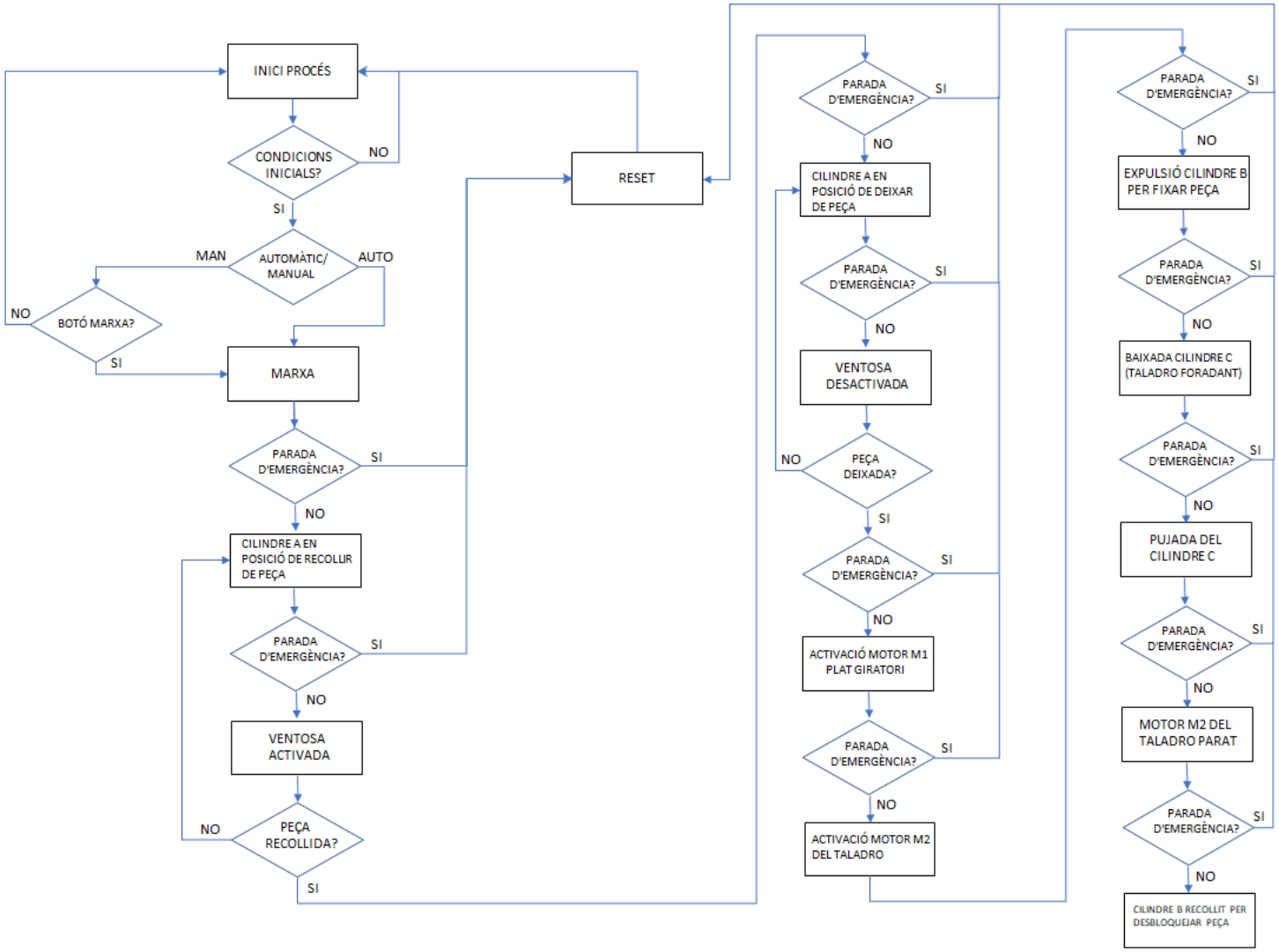


Il·lustració 62. Etapa 1-2 del programa Ladder/HMI

Un cop s'ha premut el botó Marxa, el sistema està en funcionament tal i com s'indica a l'HMI, i pel que respecta a Ladder, representa que el cilindre A0 ja s'ha mogut per recollir la peça que està esperant a la cinta d'entrada, i el programa no avançarà a l'etapa 3 fins que no s'activin la ventosa (D) com també indica l'HMI i el Sensor de Buit de la peça.

D'aquesta manera a partir d'una simulació es pot saber el correcte funcionament del programa amb Sysmac Studio.

A continuació es mostra un diagrama de flux de la seqüència lògica descrita de l'estació automatitzada:



Il·lustració 63. Diagrama de flux del funcionament de l'estació automatitza en entorn real (Laboratori)

4.5 PROGRAMACIÓ MICROSOFT EXCEL

Degut a no poder accedir a l'estació II del laboratori de Robòtica i CIM, en aquest projecte s'ha decidit utilitzar el software Microsoft Excel, i programar mitjançant Visual Basic (vba) una simulació de l'estació que es troba al laboratori, per poder veure aplicat el funcionament del reconeixement de veu.

A continuació s'explica el procés de programació d'aquesta Simulació de l'estació automatitzada.

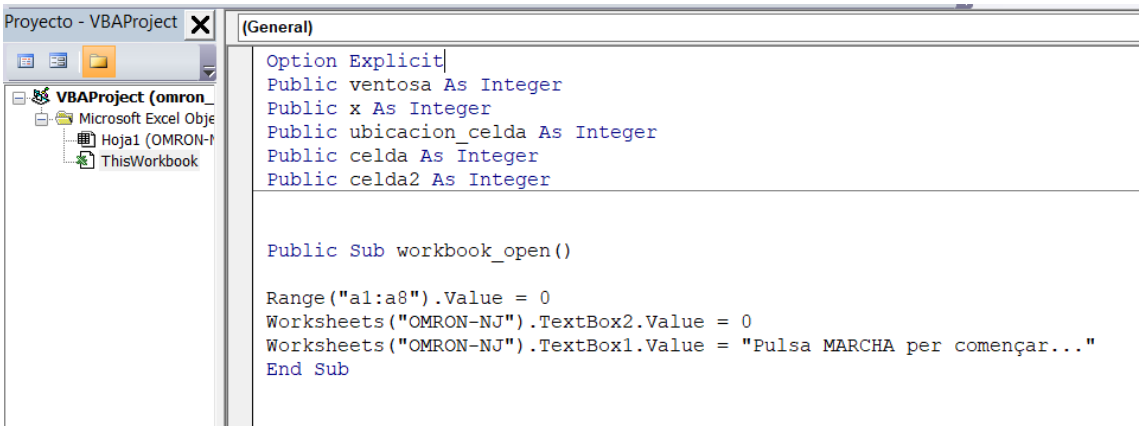
Pel fet d'haver utilitzat el software Microsoft Excel, hi ha certs aspectes que no es poden igualar al funcionament real de l'estació Automatitzada. Aquests aspectes són els següents:

- S'ha creat un botó "Nueva Pieza" per entrar peces al sistema, ja que no hi ha cap sensor funcional per indicar-ho com al sistema real de l'estació.

- Quan el procés està en mode "Automàtic" el sistema està programat per anar entrant peces sempre que una peça surti per la cinta de sortida.

- Només s'han simulat 6 sensors del sistema real del de l'estació automatitzada.

Primerament, és important executar el software Microsoft Excel com Administrador per permetre l'accés DDE al servidor DDE KEPServerEX. Un cop obert l'arxiu d 'Excel amb extensió xlsm, és a dir, un arxiu habilitat per macros, el primer que s'executa és el que hi ha dins de la subrutina "workbook_open()":



Il·lustració 64. Estat inicial Simulació Estació Automatitzada amb Microsoft Excel

La subrutina "workbook_open()" inicialitza totes les variables amb valor 0, és a dir, tot els actuadors o variables que intervenen amb la comunicació de l'Excel

amb el KEPServerEX i deixa al sistema esperant a que es premi el botó Marxa per poder començar la seqüència. A més, al (General), a part d'inicialitzar el Workbook_open(), també es declaren les variables generals del sistema.

En el programa d'Excel, hi ha una subrutina principal, la qual comportarà tota la seqüència de l'estació automatitzada esmentada en l'apartat anterior 4.3, la qual anirà cridant a les altres subrutines a mesura que es vagi desenvolupant la simulació.

A continuació es mostrarà un exemple de la subrutina principal "*Public Sub Marcha()",* on es pot veure com va cridant la resta de subrutines a mesura que es desenvolupa la seqüència del programa:

```
Public Sub Marcha()
Continua:
    Worksheets("OMRON-NJ").TextBox1.Value = "Sistema en moviment..."
    Range("A1:A2").Value = 0
    Range("A4:A6").Value = 0
    Application.ScreenUpdating = True
    Range("z25").Activate
    Range("Z25").Interior.Color = 255
    DoEvents
    ubicacion_celda = 25
For x = 0 To 10
    ActiveSheet.Shapes.Range(Array("17 Oval")).Select 'Indicador de pieza en cinta
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
    DoEvents
    Range("z25").Activate
    celda = ubicacion_celda - x
    celda2 = celda + 1
    Range("Z" & celda).Interior.Color = 255
    Range("Z" & celda2).Interior.Color = RGB(191, 194, 194) 'Color de la cinta
    Range("Z27").Interior.Color = xlNone
    Range("Z26").Interior.Color = xlNone

    DoEvents
    comprueba_Paro
    ' Activa la ventosa para mover la pieza encima del torno
    If x = 10 Then
        ActiveSheet.Shapes.Range(Array("Oval 13")).Select 'Indicador de ventosa en funcionamiento
        Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
        Application.Wait (Now + TimeValue("0:00:01"))
        Nueva_Pieza.Visible = True
        Comprueba_Nueva_Pieza
```

Il·lustració 65. Exemple Seqüència Simulació amb Microsoft Excel

En aquest cas, per exemple, es pot observar com la subrutina principal Marcha(), crida a altres subrutines com poden ser "*comprueba_Paro",* aquesta subrutina comprova si s'ha premut el botó de parada d'emergència i "*Comprueba_Nueva_Pieza"*, aquesta subrutina comprova si s'ha premut el botó de nova peça.

Les subrutines definides al programa són les següents:

- *A_M_Change()* → Realitza el canvi d'automàtic a manual, és a dir, automàtic, manualment no es pot controlar res, tot va controlat a través del reconeixement de veu, i manual els controls del programa s'han de fer manualment.

- *BTN_Reset_Click()* → Quan el botó Reset és polsat, s'executa la subrutina "Reset()" i posa en "no visible" el propi botó de Reset i el nova peça.

- *Comprueba_Nueva_Pieza()* → Comprova si durant la seqüència s'ha premut el botó "Nueva Pieza", que indica que una peça està entrant per la cinta d'entrada per ser recollida.

- *Comprueba_Paro()* → Comprova si durant la seqüència s'ha premut el botó "Parada d'Emergència", per parar tot el sistema en cas de que el botó sigui premut.

- *Marcha()* → És la subrutina més important, és la que controla tota la seqüència descrita al punt 4.3 i va cridant a les altres subrutines que intervenen en el procés de la simulació.

- *Marcha_manual_Click()* → Quan el botó Marxa es polsat, s'executa la subrutina principal "Marcha()" només quan el sistema està en Manual.

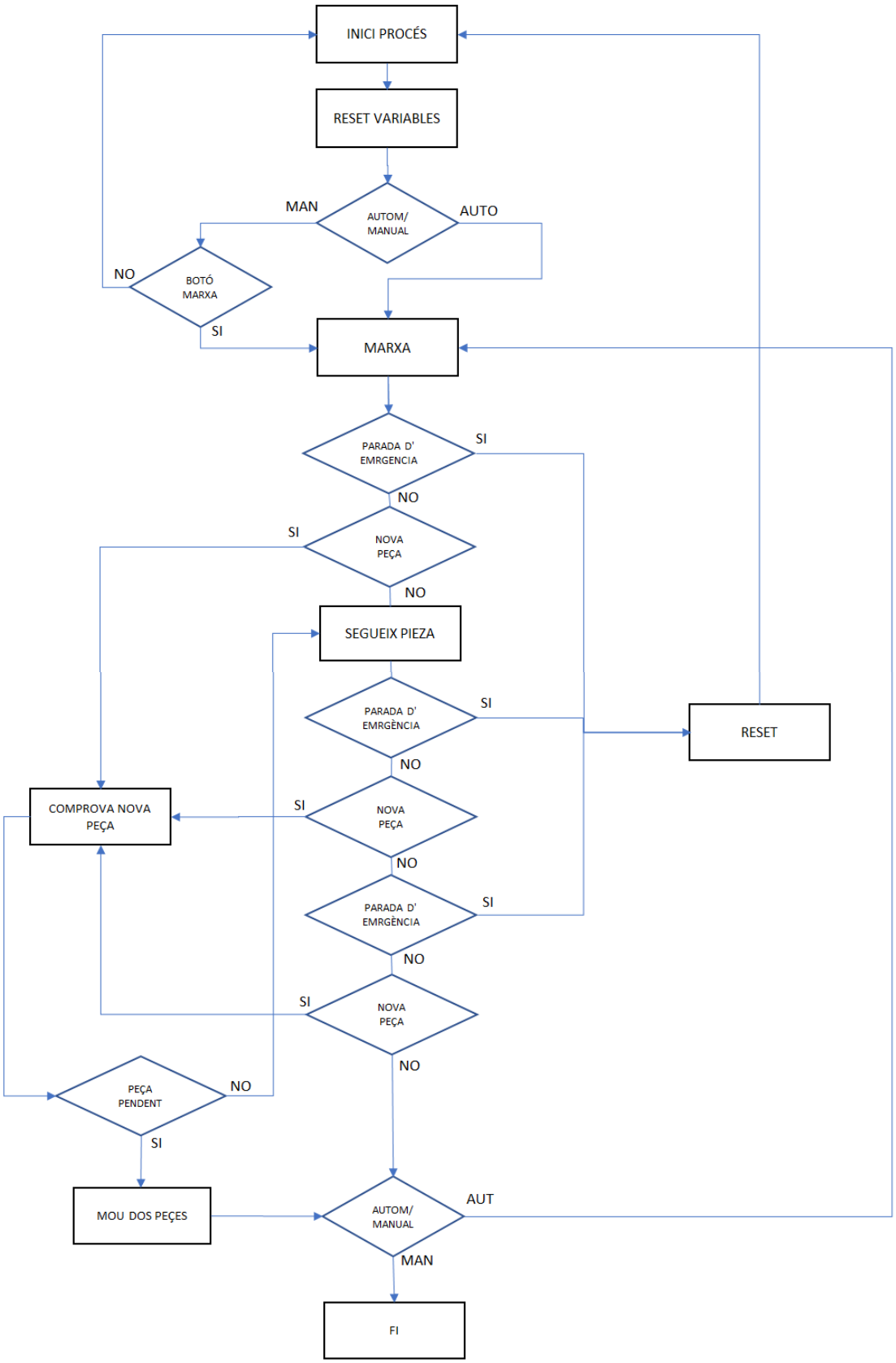
- *Mueve_dos_piezas()* → És la subrutina que es crida quan s'ha premut el botó "Nueva Pieza", ja que, al entrar una altra peça al sistema, degut al cilindre A de doble tija, s'han de desplaçar les dues peces a l'hora pel sistema.

- *Nueva_Pieza_Click()* → Quan el botó "Nueva Pieza" és polsat, s'executa la subrutina "Comprueba_Nueva_Pieza()", ja que, haurà entrat una nova peça al sistema.

- *Paro_Emergencia_Click()* → Quan el botó "Paro Emergencia" és polsat, el sistema queda parat, i es queda esperant l'ordre de "Reset" per poder inicialitzar el sistema, i deixar-ho en condicions inicials.

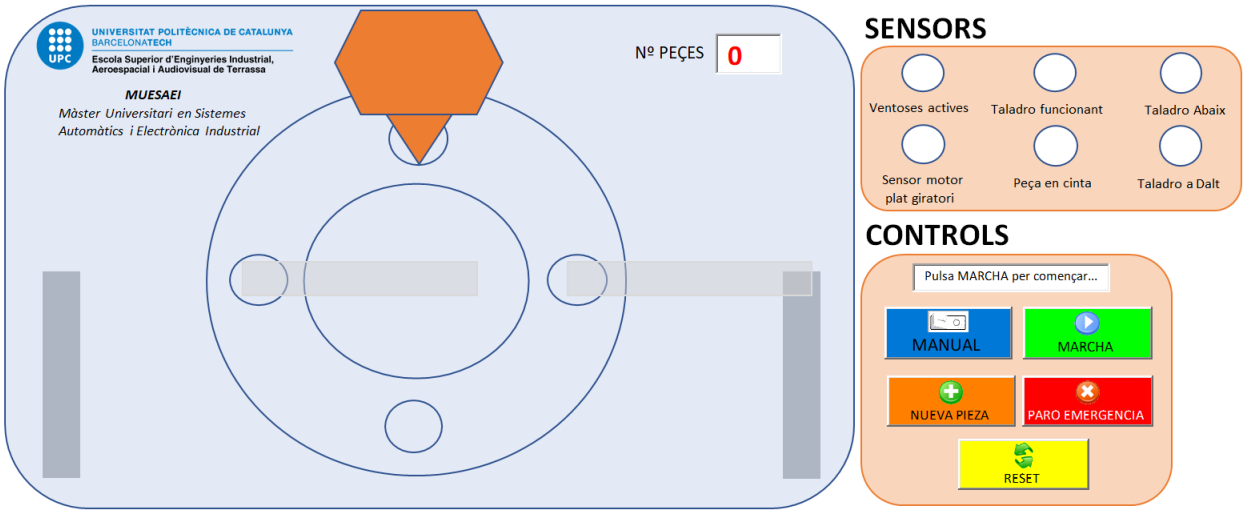
- *Reset()* → És la subrutina que torna tots els sensors, variables i controls a condicions inicials un cop s'ha premut el botó de parada d'emergència.

Segons la seqüència descrita en l'apartat 4.4 del projecte, la simulació en Microsoft Excel (Vba) presenta el següent diagrama de flux:



Il·lustració 66. Diagrama de flux del funcionament de la simulació amb Microsoft Excel (Vba)

A la il·lustració 66, es pot veure la representació de l'estació en Microsoft Excel:



Il·lustració 67. Simulació Estació automatitzada Microsoft Excel

5. PROVES I RESULTATS

Durant el desenvolupament d'aquest projecte, s'ha portat a terme el control de moviment d'una estació automatitzada a través d'una aplicació de reconeixement de veu. A continuació es detallen els problemes trobats durant el desenvolupament del projecte i les solucions donades en funció dels objectius establerts.

5.1 INTEGRACIÓ DEL SISTEMA

Tot el sistema per posar en marxa l'aplicació de control de moviment de l'estació automatitzada anirà integrat en la placa LattePanda. És necessari connectar la placa amb alimentació elèctrica per tenir energia. A la placa hi haurà connectat el teclat, el "mouse", els auriculars amb el micròfon, el cable Ethernet amb el PLC i un monitor HDMI com en un ordinador.

Per poder configurar la placa LattePanda, serà necessari afegir-li una IP que pertanyi al rang d'IP's del PLC per poder connectar-la amb el seu servidor.

Degut a no haver-hi pogut accedir al laboratori de Robòtica i CIM, no s'ha pogut provar el correcte funcionament de la placa LattePanda.

Com a solució a aquest inconvenient, s'ha comprovat el funcionament general del projecte en un ordinador amb versió Windows 10 com la placa, i el funcionament és correcte.

5.2 APLICACIÓ DE RECONEIXEMENT DE VEU

L'aplicació de reconeixement de veu s'ha creat amb el llenguatge de programació Python. El reconeixement de veu funciona correctament, ja que, es genera la senyal d'àudio i el programa transcriu a text aquesta senyal.

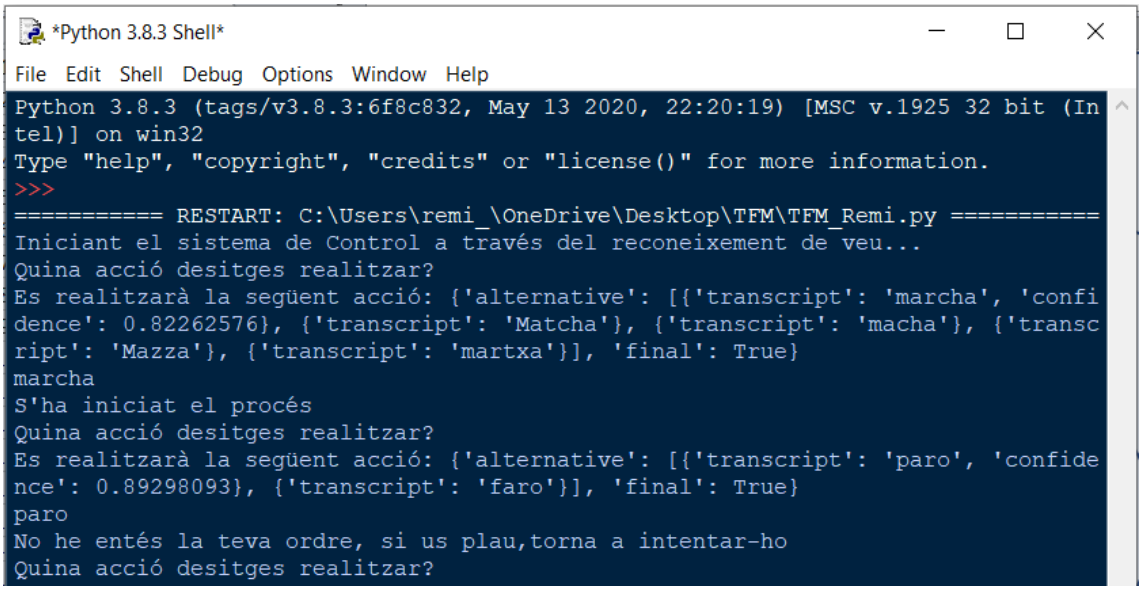
El llenguatge de reconeixement de veu que s'ha utilitzat en aquest projecte és majoritàriament el castellà, encara que també hi ha alguna instrucció en anglès. El problema dels idiomes és que si la persona o operari que l'hagi d'utilitzar no té bona pronunciació, l'aplicació pot reconèixer paraules sense sentit.

Per poder evitar aquests problemes, s'ha decidit realitzar un entrenament de les comandes d'instrucció de l'estació automatitzada. Aquest entrenament integra les respostes més comuns que l'API de Google dóna com a resposta en cada comanda.

A l'inici i durant el funcionament de l'aplicació, el programa pot escoltar qualsevol soroll de l'entorn, això pot provocar problemes de seguretat, ja que si

algú es troba a prop del micròfon pot provocar una acció en l'estació no desitjada.

Per evitar que això pugui passar s'ha creat un bucle al programa, el qual en el moment que es transcriu alguna cosa diferent de la instrucció esperada, torna a preguntar quina és l'acció desitjada, ja que no reconeix la instrucció demanada. D'aquesta manera únicament el programa respondrà correctament quan la senyal d'àudio generada transcrigui específicament la instrucció esperada.



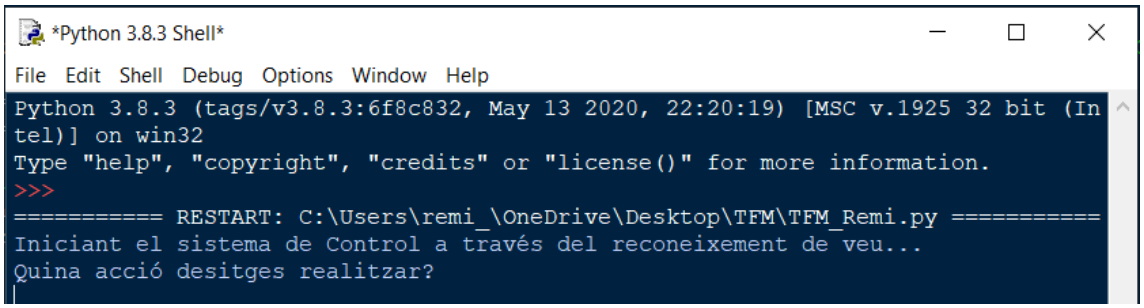
Il·lustració 68. Exemple d'instrucció no reconeguda amb reconeixement de veu

En aquest cas, rep la instrucció "Paro", com que només existeix la instrucció "Paro de Emergencia", el programa no ho reconeix com instrucció.

El soroll de l'entorn pot provocar interferències en la senyal d'àudio que es genera i això pot provocar que el programa no sigui capaç de reconèixer aquesta senyal.

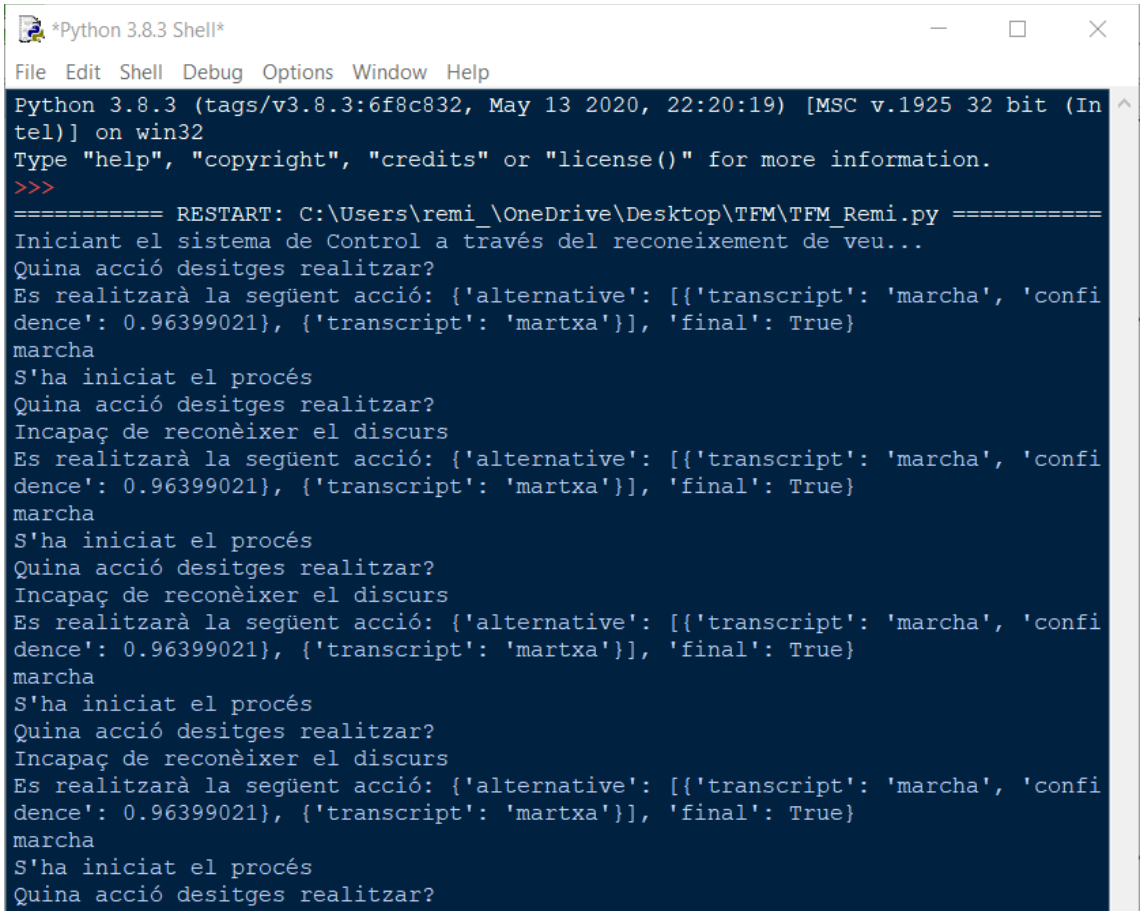
Per reduir aquestes interferències que pot provocar el soroll de l'entorn, s'ha aplicat un filtre de soroll ambiental al programa, de manera que redueix aquest soroll de l'entorn, i permet que la senyal d'àudio generada sigui millor i que l'API de Google sigui capaç de reconèixer les instruccions correctament.

Inicialment, al iniciar el sistema de control a través del reconeixement de veu, l'aplicació demana a l'usuari que digui l'acció que desitja realitzar. L'acció preferiblement, hauria de ser "Marcha" perquè el sistema inicií.



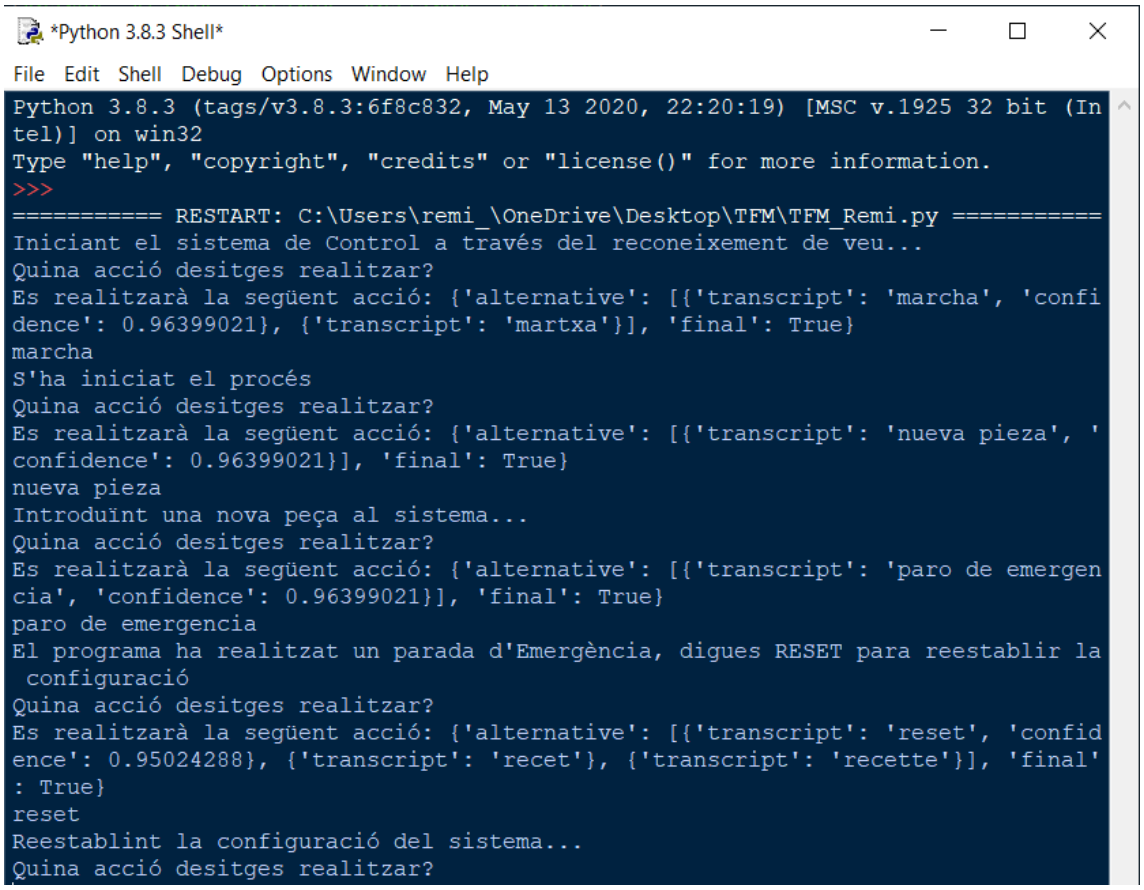
Il·lustració 69. Inici de l'aplicació de Control de reconeixement de veu

Un cop l'aplicació rep la instrucció, s'estableix comunicació amb el servidor OPC, i el programa torna a demanar una altra acció a desitjar, de manera que està constantment preguntant una nova acció. En cas de no rebre cap instrucció, torna a fer la pregunta realitzant la última acció que se li havia demanat.



Il·lustració 70. Exemple de no rebre cap instrucció

En la següent imatge es mostra que la comanda "Marcha" inicia el procés del sistema, la comanda "Nueva Pieza" introdueix una nova peça al sistema i les comandes "Parada de Emergencia" y "Reset", paren el sistema i reestableixen la configuració inicial de l'estació Automatitzada.



Il·lustració 71. Seqüència exemple del sistema de reconeixement de veu

També existeixen les instruccions "Terminar" i "Salir". "Terminar" serveix per realitzar les útlimes peces que queden al procés, i que l'estació estigui esperant l'ordre "Marxa" per tornar a iniciar el sistema. En canvi, "Salir" serveix perquè l'usuari surti del sistema i es tanqui l'aplicació de reconeixement de veu.

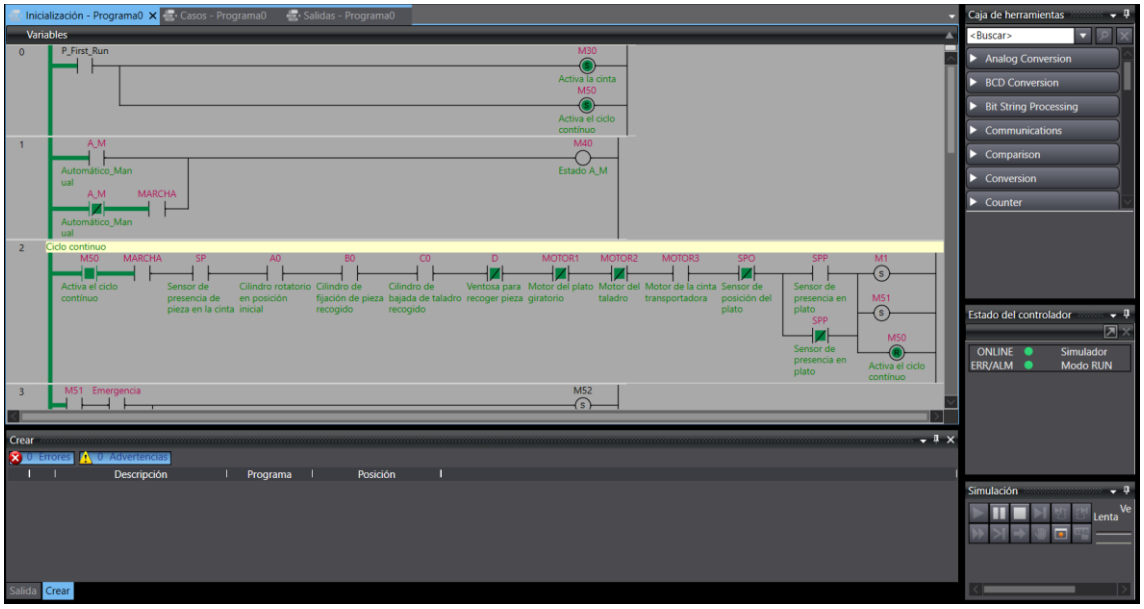
L'inferfície del programa és simple i senzilla, però bastant funcional. S'obtenen bons resultats, i es veu com el sistema respòn amb les comandes demanades amb la parla.

5.3 ESTACIÓ AUTOMATITZADA EN ENTORN REAL (LABORATORI)

Degut a la pandèmia ocasionada pel COVID-19, no s'ha pogut accedir a l'estació II del laboratori de Robòtica i CIM, per tant, no s'ha pogut comprovar el funcionament real de l'estació.

El reconeixement de veu canvia el valor de les variables del servidor OPC, per tant, teòricament també hauria de canviar el valor de la variable del software Sysmac Studio.

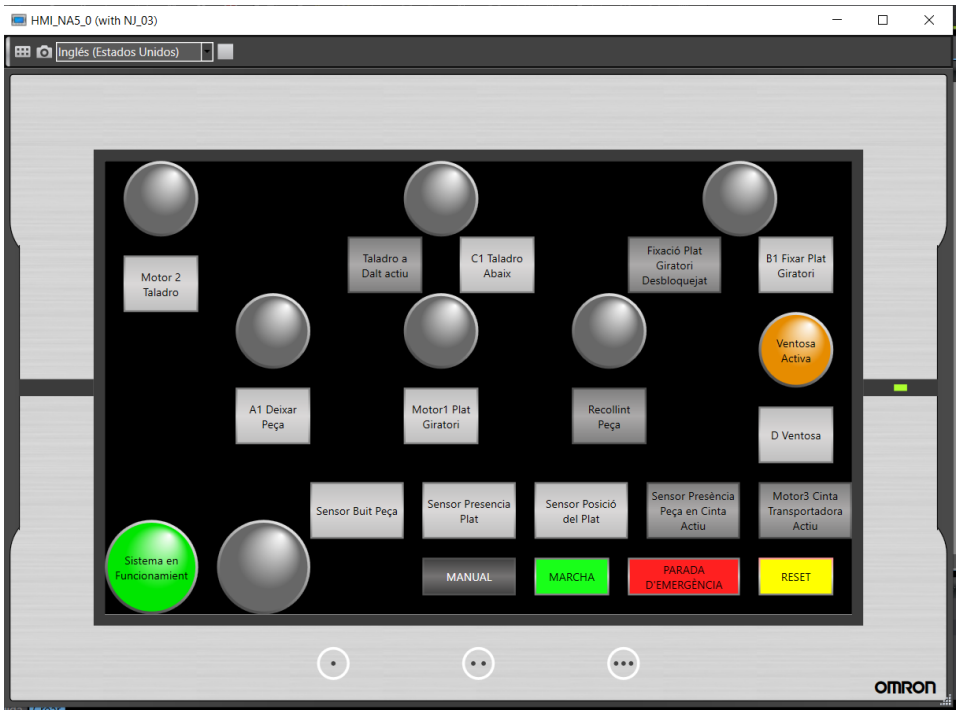
El programa creat pel PLC no genera cap error, i com a mètode de comprovació del programa creat pel PLC, s'ha utilitzat la simulació del programa Sysmac Studio. Amb aquesta simulació es pot seguir el curs de la seqüència del procés de l'estació automatitzada.



Il·lustració 72. Exemple de Simulació Sysmac Studio amb cap error generat

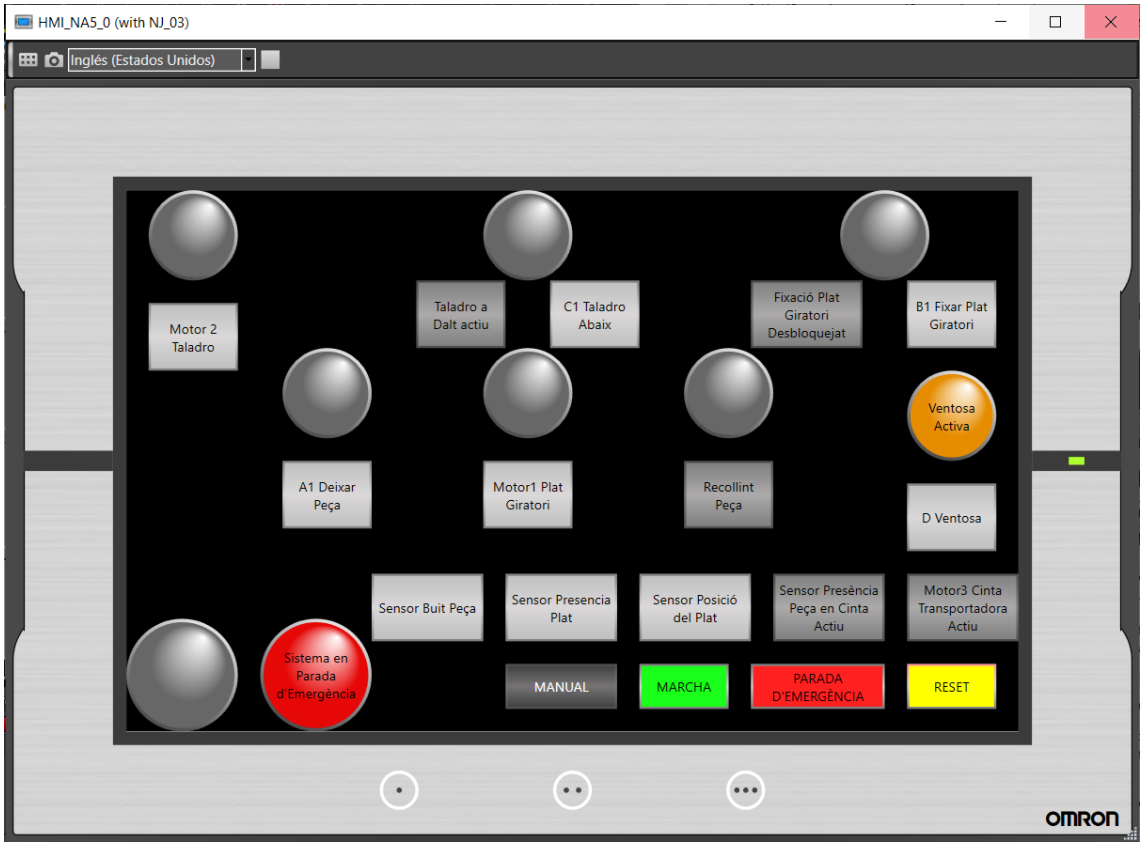
Els problemes que poden originar és que encara que el programa no generi cap error, és necessari comprovar-ho en entorn real per poder veure els temps de seqüència, i en definitiva poder veure com reacciona el sistema quan els botons "Marxa", "Parada d'emergència", "Reset" i "Automàtic/Manual" són pressionats.

Per evitar això, s'ha fet una comprovació més exhaustiva, és a dir, s'ha creat un HMI del sistema de manera que les entrades del sistema estan representades amb botons que simulen els sensors de l'estació i els botons "Marxa", "Parada d'emergència", "Reset" i "Automàtic/Manual". Pel que fa a les sortides, són representades per indicadors lluminosos els quals simulen els moviments dels cilindres, del plat giratori i dels motors del sistema.



Il·lustració 73. Simulació sistema HMI en funcionament

En la següent imatge es mostra quan el sistema està en parada d'emergència i espera l'ordre de reset per tornar a l'estat inicial del sistema.



Il·lustració 74. Simulació sistema HMI en parada d'emergència

5.4 SIMULACIÓ ESTACIÓ AUTOMATITZADA AMB MICROSOFT EXCEL (VBA)

Com alternativa de comprovació de l'aplicació de reconeixement de veu aplicada a un procés, s'ha creat una simulació de l'estació automatitzada amb el software Microsoft Excel amb Visual Basic.

El funcionament de la simulació es correcte, ja que realitza tot el procés requerit per l'estació real del laboratori de Robòtica i CIM.

El principal inconvenient que ha generat el fet de programar amb Visual Basic, és que la seqüència establerta d'instruccions de la programació, és més ràpida que la percepció de la visualització de l'usuari.

Per evitar això, s'ha hagut d'afegir les instruccions "Do Events" i "Temps d'espera" per igualar el temps de la seqüència de la programació amb la percepció de la visualització de l'usuari.

5.5 COMUNICACIÓ AMB SERVIDOR OPC

Com s'ha esmentat anteriorment, el sistema de comprovació del correcte funcionament de l'aplicació de reconeixement de veu ha sigut a través de la simulació creada amb el software Microsoft Excel amb Visual Basic.

El protocol de comunicació utilizat amb el servidor OPC DDE funciona correctament, ja que l'aplicació implementada amb Python proporciona informació del moviment a realitzar de l'estació i la simulació ho executa correctament.

Va ser necessari establir la comunicació entre l'aplicació de reconeixement de veu amb el programa Python i el servidor OPC a través del software KEPServerEX, i després comunicar aquest servidor OPC amb la Simulació creada de l'estació automatitzada amb el software Microsoft Excel.

Al connectar el Python amb el servidor OPC a través del OpenOPC amb arquitectura de 64 bits generava un error de sistema, el qual no permetia visualitzar els servidors disponibles, per tant, no es podia connectar amb el servidor OPC a través de KEPServerEX.

Com a solució, es va decidir de descarregar i instal·lar un altre cop les llibreries, el mateix software Python, els drivers i els arxius whl i tot el necessari per la correcta connexió entre el sistema de reconeixement de veu i el servidor OPC amb l'arquitectura de 32 bits.

Per establir la comunicació completa s'han generat quatre variables les quals són els quatre botons (Automàtic/Manual, Marxa, Parada d'emergència i Reset)

que hi ha a l'estació real. A més, per diferències entre l'estació real i la simulació, també s'ha generat la variable "Nueva Pieza", la qual simula la introducció d'una nova peça al sistema. Amb aquestes cinc variables es comunica tota la informació necessària per poder controlar el moviment de l'estació.

5.6 TREBALL FUTUR

Durant el desenvolupament del sistema mitjançant reconeixement de veu, hi han hagut alguns punts els quals podrien millorar-se per optimitzar el sistema.

Primerament, com a proposta de millora seria poder probar aquest projecte en una estació automatitzada real, de manera que es podria veure reflexat el treball realitzat i en cas del correcte funcionament, poder utilitzar aquesta tecnologia per processos industrials reals.

Al iniciar el sistema de control a través del reconeixement de veu actualment no demana cap instrucció per iniciar el sistema. L'aplicació demana directament l'acció a realitzar, d'aquesta manera qualsevol persona, operari o soroll, podria interrompre en el seu ús. Una proposta de millora per aquest sistema és que l'aplicació demani a l'usuari que digui la instrucció "Iniciar sistema de reconocimiento de voz", d'aquesta manera l'usuari s'assegura que quan hagi d'utilitzar l'aplicació cap persona o soroll ho podria interrompre.

L'ús de l'API de Google funciona molt bé i correctament, però té l'inconvenient que el temps de resposta del servidor triga en analitzar una ordre entre 5 i 10 segons. Això genera una mica de retard en el moment en que l'usuari executa una instrucció i aquesta es veu reflexada en l'estació automatitzada. Per tant, una millora en aquest temps de resposta seria utilitzar mètodes de reconeixement de veu, els quals no requereixin de serveis de processament externs.

5.7 IMPACTE MEDIAMBIENTAL

Aquest projecte no implica cap impacte mediambiental significatiu en quant al seu desenvolupament i implementació. En tot cas, si l'equip d'aquesta estació automatitzada fòs real i estigués orientat a un procés productiu, un cop estigués obsleta, es rebutjaria en una retirada selectiva.

6. PRESSUPOST

Aquest apartat desenvolupa l'estudi dels costos econòmics de la elaboració d'aquest projecte. Per a la realització de l'estudi dels costos econòmics, es consideren els següents aspectes tenint en compte que ha sigut realitzat per un enginyer facturant les hores de treball:

- Preus del material de l'empleat
- Temps dedicat al desenvolupament de la programació
- Temps dedicat al desenvolupament de la documentació

6.1 COST MATERIAL DE L'EMPLEAT

A continuació es mostrarà els costos relacionats amb el material d'oficina, paper, programes, i als softwares utilitzats per a la realització del projecte:

Material/Software	Preu (€)
Placa Lattepanda	150
Teclat	30
"Mouse"	15
Auriculars amb micròfon	50
Microsoft Office 2010	299
Sistema operatiu Windows 10	259
TOTAL	**803 €**

Taula 4. Cost total material de l'empleat

6.2 COST DE MÀ D'OBRA

El cost de mà d'obra es calcula en funció de les hores que s'hagin empleat per l'enginyer tècnic i del cost per hora de treball. A continuació es mostren els costos per les hores dedicades al desenvolupament del projecte:

Concepte	Hores
Investigació	90 hores
Programació de l'aplicació de reconeixement de veu	50 hores
Programació del PLC	50 hores
Programació de la Simulació amb Microsoft Excel	70 hores
Proves de funcionament	40 hores
Elaboració de la documentació	115 hores
TOTAL	**415 hores**

Taula 5. Cost de mà d'obra

Amb el càlcul de cost per hora de treball i el temps total dedicat per a la realització del projecte s'obté un cost total de mà d'obra:

COST TOTAL MÀ D'OBRA	
Cost persona/hora	30 €/h
Total hores treballades	415 hores
TOTAL	**12.450 €**

Taula 6. Cost total de mà d'obra

El cost total de la mà d'obra serà de 12.450€.

6.3 COST TOTAL

Si sumem el cost total de mà d'obra amb el cost total de material de l'empleat, s'obté el cost total del projecte.

COST TOTAL	
Cost material de l'empleat	803€
Cost total mà d'obra	12.450€
TOTAL	**13.253€**

Taula 7. Cost total del projecte

El projecte de desenvolupament d'una aplicació de control d'una estació automatitzada mitjançant el reconeixement de veu té un cost final de 13.253€.

7. LIMITACIONS DEL PROJECTE

Aquest apartat tracta sobre les limitacions que han aparegut al llarg de la realització d'aquest projecte.

La limitació principal, com s'ha anat comentant en alguns apartats del projecte, ha sigut no poder accedir a l'estació II del Laboratori de Robòtica i CIM degut a la pandèmia ocasionada pel SARS-CoV-2 (COVID-19). El fet de no poder accedir al laboratori, ha contribuït en no poder veure el funcionament real de l'estació, i en no poder veure aplicat en un entorn real l'aplicació del sistema de reconeixement de veu.

Degut a aquesta impossibilitat, s'han trobat alternatives, com ha sigut la de crear una simulació de l'estació real a través del software Microsoft Excel (vba), en el qual s'ha pogut veure la correcta aplicació del sistema de reconeixement de veu.

Pel fet d'haver creat una simulació amb Microsoft Excel ha comportat un temps afegit que no estava previst pel desenvolupament del projecte. Encara que finalment s'ha pogut assolir correctament i pot donar una visió acurada de l'automatització desitjada.

8. CONCLUSIONS

Malgrat les limitacions que han sorgit alienes al projecte, s'ha assolit de manera satisfactòria les dos àrees principals plantejades inicialment, que són l'automatització industrial i les comunicacions industrials en una aplicació funcional capaç de controlar el moviment d'una estació automatitzada mitjançant el reconeixement de veu.

Partint d'un coneixement inicialment bàsic en programació amb Python, s'ha pogut aprofundir i aprendre sobre el llenguatge de programació, el qual és un dels llenguatges amb major creixement en l'actualitat, i s'està començant a aplicar en empreses.

Ha sigut possible aprofundir també en el llenguatge de programació de Visual Basic, ja que s'ha hagut de crear una simulació de l'estació automatitzada a través de Microsoft Excel, el qual també s'ha assolit satisfactòriament, ja que la simulació creada emula correctament la seqüència requerida per l'estació II del laboratori de Robòtica i CIM.

Partint d'un coneixement molt bàsic sobre el reconeixement de veu, s'ha aconseguit comprendre el seu funcionament, com es pot utilitzar i el seu ús aplicat a diferents àmbits.

Les proves de funcionament del reconeixement de veu no s'han pogut realitzar en un entorn real amb soroll ambiental afegit, però inclús amb la simulació creada a través de Microsoft Excel ha sigut possible demostrar que és una aplicació funcional.

Gràcies a aquest projecte s'ha pogut aprofundir en temes d'automatització industrial amb programació de PLC i comunicació OPC en dos sentits diferents aplicat a un entorn real i aplicat a una simulació a través de Microsoft Excel. Finalment, s'ha familiaritzat amb l'entorn de simulació de Sysmac Studio mitjançant un desenvolupament HMI del sistema.

9. BIBLIOGRAFIA

Laca, M., *Qué es Python – Definición, características y ventajas.* Pythones. Recuperat el 25 de Maig de 2020 de https://pythones.net/que-es-python-y-sus-caracteristicas/#%C2%BFQue_es_Python

Robledano, A. (23 de Setembre de 2019). *Qué es Python: Características, evolución y futuro*. OpenWebinars. Recuperat el 25 de Maig de 2020 de https://openwebinars.net/blog/que-es-python/

ABC (16 de Febrer de 2015). *¿Qué es una API y para qué sirve?*. ABC consultorio. Recuperat el 26 de Maig de 2020 de https://www.abc.es/tecnologia/consultorio/20150216/abci--201502132105.html

Suarez, I. (28 de Novembre de 2018), *¿Qué es una API?*. Google Site. Recuperat el 26 de Maig de 2020 de https://sites.google.com/site/ivettsuarez236/que-es-google-classroom/5-2que-es-api

Barroso, J. (23 de Març de 2016). *API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos*. BBVA Api-Market**.** Recuperat el 26 de Maig de 2020 de https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos

NochesDeCode (1 de Novembre de 2011). *Introducción al lenguaje de programación Ladder*. Noches de code. Recuperat el 28 de Maig de 2020 de http://www.nochesdecode.com.ar/2011/09/introduccion-al-lenguaje-de-programacion-ladder/

Google Developers. *SPEECH-TO-TEXT*. Google Cloud. Recuperat el 29 de Maig de 2020 de https://cloud.google.com/speech-to-text?hl=es

Universitat d'enginyeria UNLP. *Diagrama de contactos (Ladder)*. Educación urbana. Recuperat el 30 de Maig de 2020 de https://www.educacionurbana.com/apuntes/ladder.pdf

Omron*. Plataforma de automatización Sysmac.* Omron. Recuperat el 2 de Juny de 2020 de https://cpi.com.ar/info_productos/pdf/NJ_Plataforma%20Sysmac.pdf

Omron. *Sysmac Studio*. Industrial Omron. Recuperat el 2 de Juny de 2020 de https://industrial.omron.es/es/products/sysmac-studio#features

AS, R. (31 de Maig de 2017) *Comunicación OPC con KEPServerEX.* Automatització Industrial. Recuperat el 3 de Juny de 2020 de http://automatizacioncavanilles.blogspot.com/2017/05/comunicacion-opc-kepserverex.html

Logitek Team (8 de Maig de 2019). *Qué es OPC y qué es un OPC Server.* Kepware Kepserverex. Recuperat el 3 de Juny de 2020 de https://www.kepserverexopc.com/que-es-opc-y-que-es-un-opc-server/

Korsan, S. *Microsoft Excel.* W3e Academy. Recuperat el 4 de Juny de 2020 de https://www.es.w3ki.com/excel/

Porto Pérez, J. *Definición de PLC.* Definición.de. Recuperat el 10 de Juny de 2020 de https://definicion.de/plc/

Amos, D. (2018). *The ultimate Guide to Speech Recognition with Python*. Real Python. Recuperat el 13 de Juny de 2020 de https://realpython.com/python-speech-recognition/

Zhang, A. (5 de Desembre de 2017). *SpeechRecognition.* PyPi. Recuperat el 13 de Juny de 2020 de https://pypi.org/project/SpeechRecognition/

Kominek, D. (Any 2009). *OPC: ¿De qué se trata, y cómo funciona?.* MatrikonOPC. Recuperat el 15 de Juny de 2020 de https://www.interempresas.net/FeriaVirtual/Catalogos_y_documentos/220446/Guia-para-entender-la-tecnologia-OPC.pdf

Cursos Ingenieros Industriales (Any 2018). *Características y versiones de OPC DA.* VITC. Recuperat el 15 de Juny de 2020 de http://www.cursosingenieriaindustrial.com/opc-da/

Cursos de Ingeniería Industrial (31 de Maig de 2018). *Caracterísitcas y usos del protocolo OPC UA en SCADA.* Recuperat el 16 de Juny de 2020 de https://vestertraining.com/caracteristicas-opc-ua-scada/

Moffitt, C. (2 de Juliol de 2018). *Automating Windows Applications Using COM.* Practical Bussiness Python. Recuperat el 16 de Juny de 2020 de https://pbpython.com/windows-com.html

Batchelor, D. (31 de Maig de 2018). *Component Object Model (COM).* Microsoft. Recuperat el 16 de Juny de 2020 de https://docs.microsoft.com/es-es/windows/win32/com/component-object-model--com--portal?redirectedfrom=MSDN

Graybox Software Developers. *OPC DA Auto Wrapper.* Graybox Software. Recuperat el 16 de juny de 2020 de http://gray-box.net/daawrapper.php?lang=en

SourceForge Developers. *OpenOPC for Python.* Source Forge. Recuperat el 16 de Juny de 2020 de http://openopc.sourceforge.net/

Kepware Developers (Febrer de 2019). *Connectivity Guide KEPServerEX, DDE and Excel.* PTC, Inc. ©2015-2019, Ref. 1.005

Carril, M. (Agost de 2018). *Guía de Comunicación por OPC UA con KEPServerEX.* Logitek real time solutions. Recuperat el dia 19 de Juny de 2020 de https://www.kepserverexopc.com/wp-content/uploads/2019/06/TNLK037-Guia-de-comunicacion-por-OPC-UA-con-KEPServerEX.pdf

Cardenas, R. (18 de Juliol de 2009). *Reconocimiento de voz.* Blogger. Recuperat el 20 de Juny de 2020 de http://reconocimientodevozz.blogspot.com/2009/07/reconocimiento-de-voz.html#:~:text=El%20objetivo%20de%20la%20tecnolog%C3%ADa,el%20uso%20de%20las%20computadoras.

Ahuactzin, A. (Any 1999). *Sistemas de reconocimiento de voz y síntesis de voz.* Tesis Licenciatura, Ingeniería en Sistemas Computacionales, Departamento de Ingenería en Sistemas Computacionales, Escuela de Ingeniería, Universidad de las Américas-Puebla. Recuperat de http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/ahuactzin_l_a/capitulo1.pdf

Campos, D. (21 de Març de 2018). *¿Cómo funciona el reconocimiento de voz automático del habla?*. Medium. Recuperat el 21 de Juny de 2020 de https://medium.com/soldai/c%C3%B3mo-funciona-el-reconocimiento-autom%C3%A1tico-del-habla-eb038ecfe72e

LattePanda (01 de Juny de 2019). *LattePanda 4G/64GB.* LattePanda. Recuperat el 23 de Juny de 2020 de https://www.lattepanda.com/products/3.html

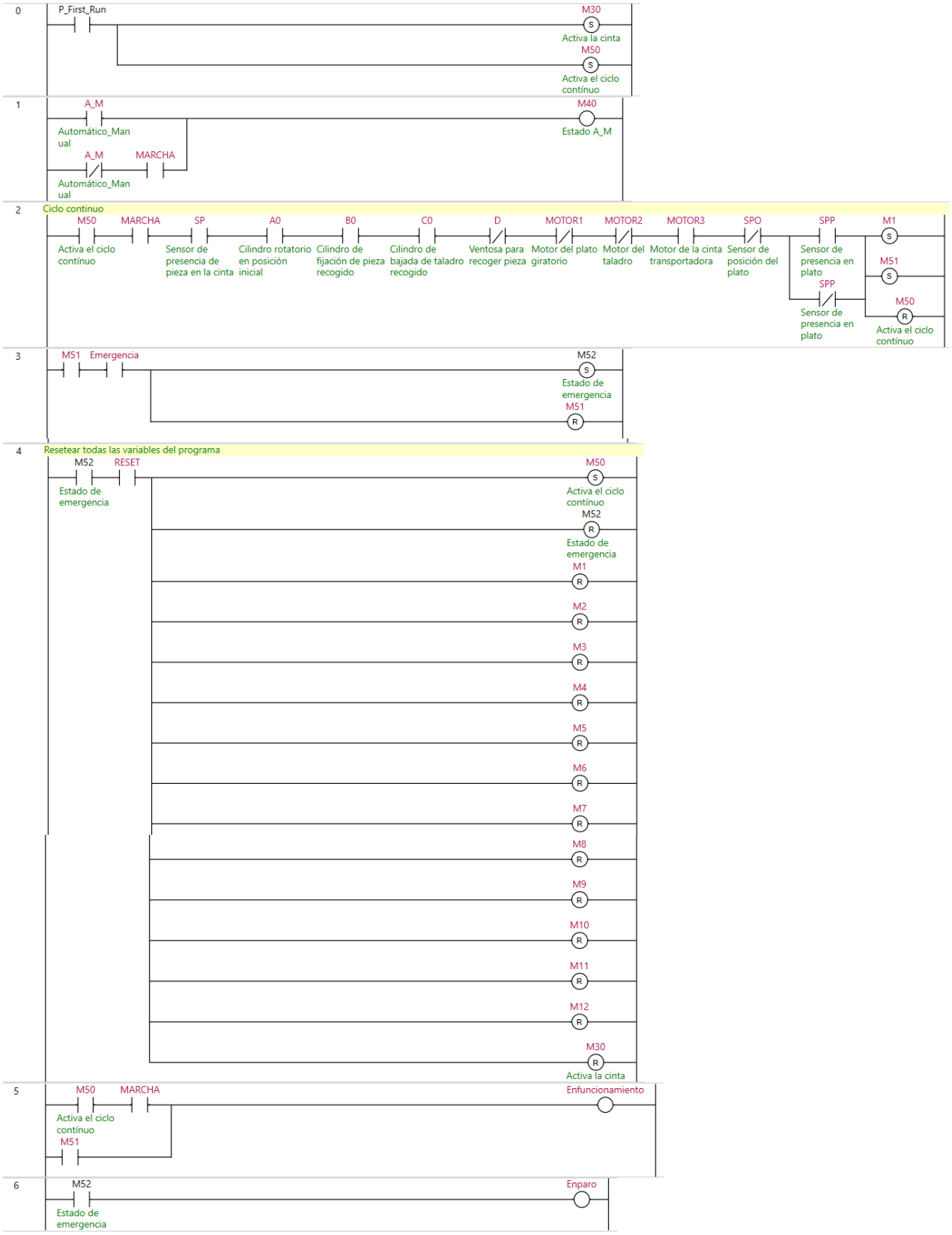
Rouse, M. (Març de 2010). *Dynamic Data Exchange (DDE).* TechTarget. Recuperat el 23 de Juny de 2020 de https://whatis.techtarget.com/definition/Dynamic-Data-Exchange-DDE

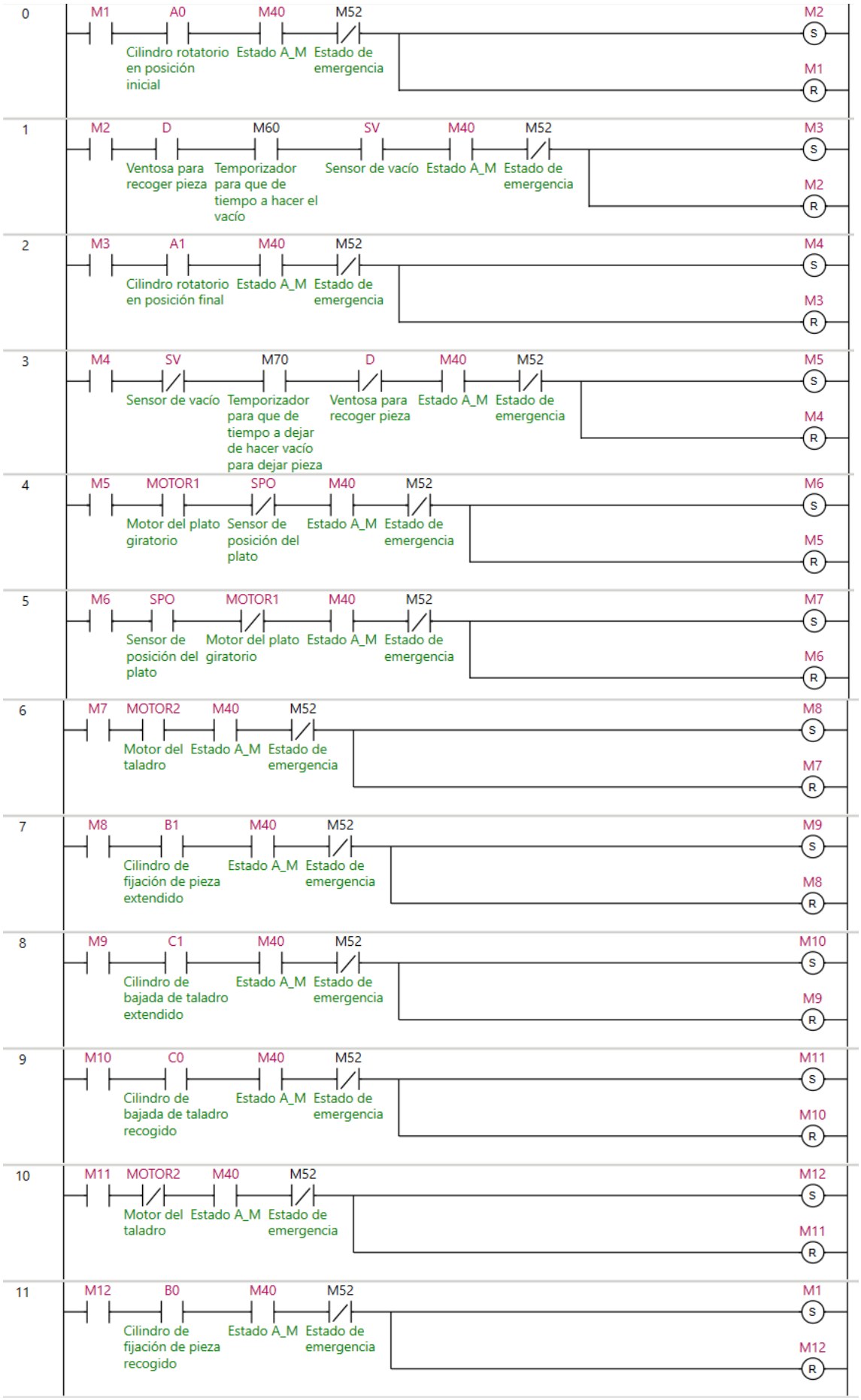
Vizcuña, J., Arrizibita, I., Unzué, J. (21 de Maig de 2019) *La importancia de la automatización de los procesos industriales.* Kuzu. Recuperat el 23 de Juny de 2020 de https://kuzudecoletaje.es/la-importancia-de-la-automatizacion-de-los-procesos-industriales/

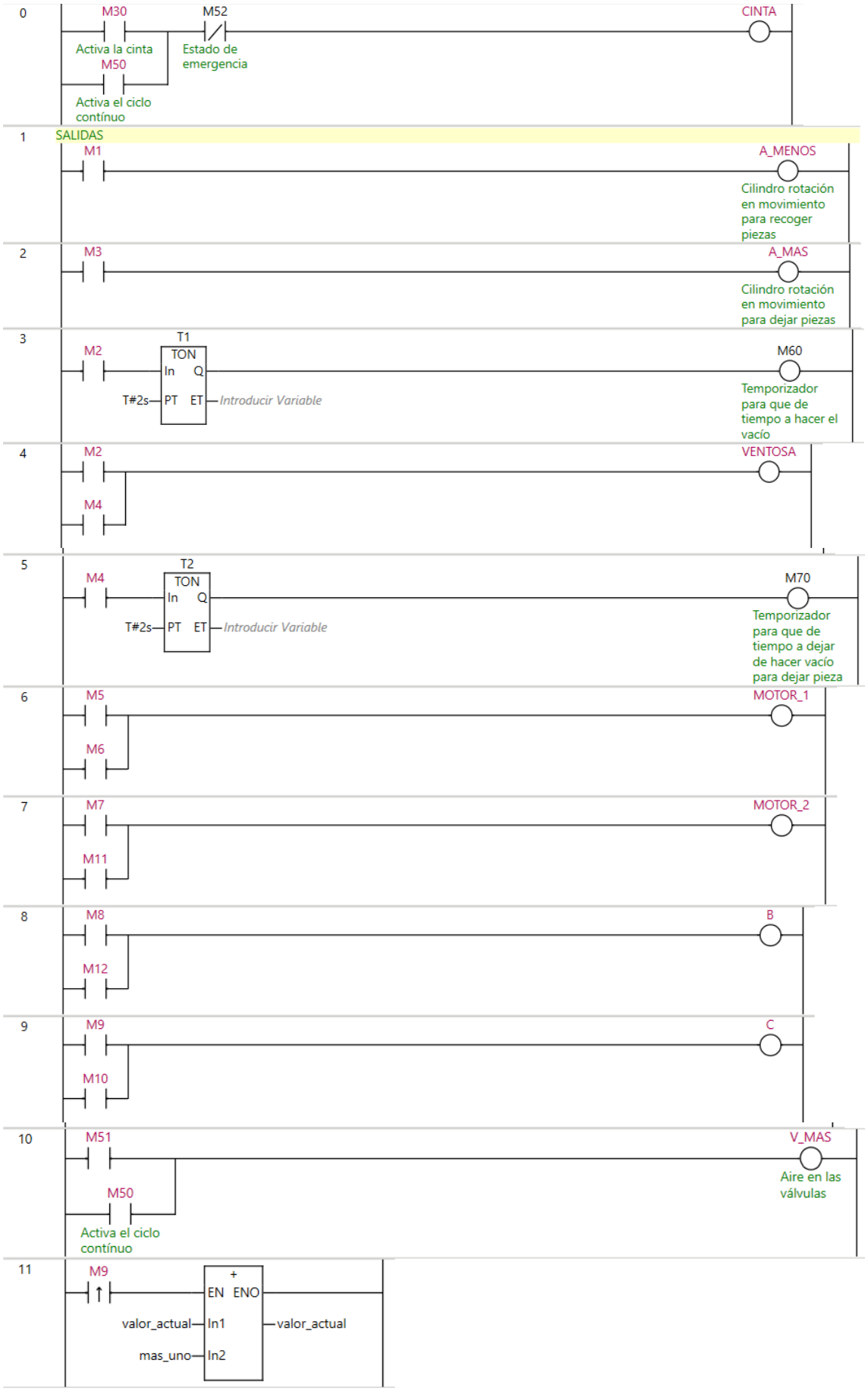
Automática e Instrumentación (Març del 2020) *Comunicaciones industriales: la propuesta para los estándares y protocolos de la nueva era*. Automática e Instrumentación. Recuperat el 23 de Juny de 2020 de http://www.automaticaeinstrumentacion.com/es/notices/2020/04/comunicaciones-industriales-la-propuesta-para-los-estandares-y-protocolos-de-la-nueva-era-46433.php#.XvIdRWgzZPY

10. ANNEXOS

10.1 ANNEX 1: CODI DE PROGRAMACIÓ DEL PLC OMRON-NJ AMB SYSMAC STUDIO




0
M1
A0
M40
M52
M2
Cilindro rotatorio en posición inicial
Estado A_M
Estado de emergencia
S
M1
R
1
M2
D
M60
SV
M40
M52
M3
Ventosa para recoger pieza
Temporizador para que de tiempo a hacer el vacío
Sensor de vacío
Estado A_M
Estado de emergencia
S
M2
R
2
M3
A1
M40
M52
M4
Cilindro rotatorio en posición final
Estado A_M
Estado de emergencia
S
M3
R
3
M4
SV
M70
D
M40
M52
M5
Sensor de vacío
Temporizador para que de tiempo a dejar de hacer vacío para dejar pieza
Ventosa para recoger pieza
Estado A_M
Estado de emergencia
S
M4
R
4
M5
MOTOR1
SPO
M40
M52
M6
Motor del plato giratorio
Sensor de posición del plato
Estado A_M
Estado de emergencia
S
M5
R
5
M6
SPO
MOTOR1
M40
M52
M7
Sensor de posición del plato
Motor del plato giratorio
Estado A_M
Estado de emergencia
S
M6
R
6
M7
MOTOR2
M40
M52
M8
Motor del taladro
Estado A_M
Estado de emergencia
S
M7
R
7
M8
B1
M40
M52
M9
Cilindro de fijación de pieza extendido
Estado A_M
Estado de emergencia
S
M8
R
8
M9
C1
M40
M52
M10
Cilindro de bajada de taladro extendido
Estado A_M
Estado de emergencia
S
M9
R
9
M10
C0
M40
M52
M11
Cilindro de bajada de taladro recogido
Estado A_M
Estado de emergencia
S
M10
R
10
M11
MOTOR2
M40
M52
M12
Motor del taladro
Estado A_M
Estado de emergencia
S
M11
R
11
M12
B0
M40
M52
M1
Cilindro de fijación de pieza recogido
Estado A_M
Estado de emergencia
S
M12
R


0
M30
Activa la cinta
M50
Activa el ciclo contínuo
M52
Estado de emergencia
CINTA
1
SALIDAS
M1
A_MENOS
Cilindro rotación en movimiento para recoger piezas
2
M3
A_MAS
Cilindro rotación en movimiento para dejar piezas
3
M2
T1
TON
In Q
T#2s PT ET
Introducir Variable
M60
Temporizador para que de tiempo a hacer el vacío
4
M2
M4
VENTOSA
5
M4
T2
TON
In Q
T#2s PT ET
Introducir Variable
M70
Temporizador para que de tiempo a dejar de hacer vacío para dejar pieza
6
M5
M6
MOTOR_1
7
M7
M11
MOTOR_2
8
M8
M12
B
9
M9
M10
C
10
M51
M50
Activa el ciclo contínuo
V_MAS
Aire en las válvulas
11
M9
+
EN ENO
valor_actual
In1
valor_actual
mas_uno
In2

10.2 ANNEX 2: CODI DE PROGRAMACIÓ AMB PYTHON

```
#########################################################################################################
#                                          ESEIAAT
#                                  TREBALL FI DE MÀSTER
#                            ESTUDI I IMPLEMENTACIÓ DEL CONTROL
#                  D'UNA ESTACIÓ AUTOMATITZADA MITJANÇANT EL RECONEIXEMENT DE VEU
#########################################################################################################

#Llibreríes necessaries per la implementació del codi
import random
import time
import speech_recognition as sr
import OpenOPC

#Ordres de veu necessàries per la execució del programa
marcha = ["marcha", "martxa", "mancha"]
paro = ["paro", "faro", "claro", "pájaro", "tarot"]
paro_emergencia = ["emergencia", "emergencias", "paro de emergencia", "paro emergencia"]
reset = ["reset", "recet", "recette", "receta", "receptor", "raclette"]
nueva_pieza = ["pieza", "empieza", "limpieza", "nueva pieza", "nueva pista", "nueva vista", "Cieza" ]
salir = ["salir"]
continúa = ["continua", "continúa", "kontinua", "qontinua"]
terminar = ["terminar", "terminal", "termita"]

#Establint comunicació amb el servidor OPC de KEPServer
opc = OpenOPC.client()
opc.connect('Kepware.KEPServerEX.V6')


def programa_iniciat():
    print("Iniciant el sistema de Control a través del reconeixement de veu...")
    return

programa_iniciat()

inici = True
while inici:

    r = sr.Recognizer()
    with sr.Microphone() as source:
        print('Quina acció desitges realitzar?')
        r.adjust_for_ambient_noise(source)
        audio = r.listen(source) #timeout=5,phrase_time_limit=10)
    try:
        respuesta = r.recognize_google(audio,language='es-ES, en-US, en-GB')
        recognized = r.recognize_google(audio,show_all=True,language='es-ES, en-US, en-GB')


    except sr.RequestError:
        print('Aplicació no disponible')

    except sr.UnknownValueError:
        print('Incapaç de reconèixer el discurs')


    print('Es realitzarà la següent acció:', recognized)
    print (respuesta)
    print


#Instruccions de veu necessàries

    if  respuesta in ["marcha", "martxa", "mancha", "hacha"]:
        print("S'ha iniciat el procés")
        opc.write(('Channel1.Device1.MARCHA', 1))
        time.sleep(3)


    elif respuesta in ["emergencia", "emergencias", "paro de emergencia", "paro emergencia"]:
        print("El programa ha realitzat un parada d'Emergència, digues RESET para reestablir la configuració")
        opc.write( ('Channel1.Device1.PARO_EMERGENCIA', 1))
```

```
    elif respuesta in ["reset", "recet", "recette", "receta", "receptor", "raclette"]:
        print('Reestablint la configuració del sistema...')
        opc.write(('Channel1.Device1.RESET', 1))
        time.sleep(1)
        opc.write(('Channel1.Device1.RESET', 0))
        opc.write(('Channel1.Device1.PARO_EMERGENCIA',0))
        opc.write( ('Channel1.Device1.NUEVA_PIEZA', 0))
        opc.write( ('Channel1.Device1.MARCHA', 0))
        opc.write( ('Channel1.Device1.PARO', 0))

    elif respuesta in ["pieza", "empieza", "limpieza", "nueva pieza", "nueva pista", "nueva vista", "Cieza" ]:
        print("Introduïnt una nova peça al sistema...")
        opc.write( ('Channel1.Device1.NUEVA_PIEZA', 1))
        time.sleep(1)

    elif respuesta in ["terminar", "terminal", "termita"]:
        print("El procés terminarà després d'acabar l'última peça, esperant nova instrucció...")
        opc.write( ('Channel1.Device1.MARCHA', 0))
        time.sleep(1)

    elif respuesta in 'salir':
        inici = False
        opc.close()

    else:
        print('No he entés la teva ordre, si us plau,torna a intentar-ho')
else:
    print("Tancant el sistema de Control a través del reconeixement de veu...")
    exit()
```

10.3 ANNEX 3: CODI DE PROGRAMACIÓ AMB MICROSOFT EXCEL VBA

```
Public Sub Marcha_manual_Click()
Marcha
Application.Wait (Now + TimeValue("0:00:01"))
End Sub
```

```
Public Sub Paro_Emergencia_Click()
'Botón paro de emergencia, esto es el valor que leerá el kepserver
Range("A5").Value = 1
BTN_Reset.Visible = True
Worksheets("OMRON-NJ").TextBox1.Value = "Esperant ordre de Reset..."
End
End Sub
```

```
Public Sub BTN_Reset_Click()
'Botón reset, pone en no visible los botones reset y nueva pieza
Reset
BTN_Reset.Visible = False
Nueva_Pieza.Visible = False
End Sub
```

```
Public Sub Marcha()
Continua:
    Worksheets("OMRON-NJ").TextBox1.Value = "Sistema en moviment..."
    Range("A1:A2").Value = 0
    Range("A4:A6").Value = 0
    Application.ScreenUpdating = True
    Range("z25").Activate
    Range("Z25").Interior.Color = 255
    DoEvents
    ubicacion_celda = 25
For x = 0 To 10
    ActiveSheet.Shapes.Range(Array("17 Oval")).Select 'Indicador de pieza en cinta
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
    DoEvents
    Range("z25").Activate
    celda = ubicacion_celda - x
    celda2 = celda + 1
    Range("Z" & celda).Interior.Color = 255
    Range("Z" & celda2).Interior.Color = RGB(191, 194, 194) 'Color de la cinta
    Range("Z27").Interior.Color = xlNone
    Range("Z26").Interior.Color = xlNone

    DoEvents
    comprueba_Paro
   ' Activa la ventosa para mover la pieza encima del torno
   If x = 10 Then
       ActiveSheet.Shapes.Range(Array("Oval 13")).Select 'Indicador de vectosa en funcionamiento
       Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
       Application.Wait (Now + TimeValue("0:00:01"))
       Nueva_Pieza.Visible = True
       Comprueba_Nueva_Pieza
       DoEvents
       ActiveSheet.Shapes.Range(Array("Group 10")).Select 'Movimiento del cilindro A para mover las piezas de la cinta al plato
       Selection.ShapeRange.IncrementLeft -150
       Range("a1").Value = 1
       ActiveSheet.Shapes.Range(Array("17 Oval")).Select 'Indicador de pieza en la cinta
       Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
   End If
   Application.Wait (Now + TimeValue("0:00:01"))
Next

sigue_pieza:
    comprueba_Paro
    Comprueba_Nueva_Pieza
    If Range("a2").Value = 1 Then
        DoEvents
        ActiveSheet.Shapes.Range(Array("Group 10")).Select 'Movimiento del cilindro A para mover las piezas de la cinta al plato
        Selection.ShapeRange.IncrementLeft -150
        Range("A1").Value = 1
        Range("A2").Value = 0
    End If
```

```
Range("Z" & celda).Interior.Color = RGB(191, 194, 194)  'Color de la cinta


Application.Wait (Now + TimeValue("0:00:01"))
DoEvents
Range("T15").Activate
ActiveSheet.Shapes.Range(Array("Oval 13")).Select 'Indicador de vectosa en funcionamiento
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)

Range("T15").Interior.Color = 255
DoEvents
comprueba_Paro
Comprueba_Nueva_Pieza
'movemos las piezas a la posición original
Range("a1").Value = 0
DoEvents
Application.Wait (Now + TimeValue("0:00:01"))

ActiveSheet.Shapes.Range(Array("15 Oval")).Select 'Indicador en funcionamiento del plato (Motor 1)
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
DoEvents
ActiveSheet.Shapes.Range(Array("Group 10")).Select 'Movimiento del cilindro A para volver a la posición inicial
Selection.ShapeRange.IncrementLeft 150

ActiveSheet.Shapes.Range(Array("18 Oval")).Select  'Indicador en funcionamiento del cilindro abajo
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
'Range("P8").Activate
Application.Wait (Now + TimeValue("0:00:03"))

Range("T15").Interior.Pattern = xlNone

ActiveSheet.Shapes.Range(Array("18 Oval")).Select  'Indicador en funcionamiento del cilindro abajo
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
ActiveSheet.Shapes.Range(Array("Oval 16")).Select  'Indicador en funcionamiento del taladro
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)

Application.Wait (Now + TimeValue("0:00:01"))

ActiveSheet.Shapes.Range(Array("15 Oval")).Select 'Indicador en funcionamiento del plato (Motor 1)
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
Application.Wait (Now + TimeValue("0:00:01"))
DoEvents
Range("P8").Interior.Color = 255
DoEvents
comprueba_Paro
Comprueba_Nueva_Pieza
Application.Wait (Now + TimeValue("0:00:05"))

ActiveSheet.Shapes.Range(Array("Oval 16")).Select 'Indicador en funcionamiento del taladro
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
ActiveSheet.Shapes.Range(Array("19 Oval")).Select  'Indicador en funcionamiento del cilindro arriba
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)

DoEvents
ActiveSheet.Shapes.Range(Array("15 Oval")).Select 'Indicador en funcionamiento del plato (Motor 1)
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)

Range("P8").Interior.Pattern = xlNone

Application.Wait (Now + TimeValue("0:00:01"))
ActiveSheet.Shapes.Range(Array("19 Oval")).Select  'Indicador en funcionamiento del cilindro arriba
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
ActiveSheet.Shapes.Range(Array("15 Oval")).Select 'Indicador en funcionamiento del plato (Motor 1)
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
'Range("K15").Activate
Application.Wait (Now + TimeValue("0:00:01"))
    ActiveSheet.Shapes.Range(Array("Oval 13")).Select 'Indicador de vectosa en funcionamiento
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
Range("K15").Activate
Range("K15").Interior.Color = 255


DoEvents
comprueba_Paro
Comprueba_Nueva_Pieza
If Cells(7, 1) = 1 Then
  mueve_dos_piezas
  End
End If
```

```
    ActiveSheet.Shapes.Range(Array("Group 10")).Select
    Selection.ShapeRange.IncrementLeft -150
    Range("a1").Value = 1
    DoEvents
    Application.Wait (Now + TimeValue("0:00:01"))
    Range("F15").Activate
    Range("F15").Interior.Color = 255
        DoEvents

    Range("K15").Interior.Pattern = xlNone

    ActiveSheet.Shapes.Range(Array("Oval 13")).Select 'Indicador de vectosa en funcionamiento
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
    Application.Wait (Now + TimeValue("0:00:01"))
    comprueba_Paro
    Comprueba_Nueva_Pieza
    Range("F14").Interior.Color = RGB(255, 255, 255)
    ubicacion_fcelda = 16
For x = 0 To 9
    ActiveSheet.Shapes.Range(Array("17 Oval")).Select 'Indicador de pieza en la cinta
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
    DoEvents
    celdaf = ubicacion_fcelda + x
    celdaf2 = celdaf - 1
    Range("F" & celdaf).Interior.Color = 255
    Range("F" & celdaf2).Interior.Color = RGB(191, 194, 194)
    Application.Wait (Now + TimeValue("0:00:01"))
    comprueba_Paro
    Comprueba_Nueva_Pieza
    If x = 3 Then
        ActiveSheet.Shapes.Range(Array("Group 10")).Select
        Selection.ShapeRange.IncrementLeft 150
        Range("a1").Value = 0
End If
Next
Range("A8").Value = Range("A8").Value + 1
Worksheets("OMRON-NJ").TextBox2.Value = Range("A8").Value
DoEvents
Range("F25").Interior.Color = RGB(191, 194, 194)
Range("F26").Interior.Color = RGB(255, 255, 255)
Range("F27").Interior.Color = xlNone
ActiveSheet.Shapes.Range(Array("17 Oval")).Select 'Indicador de pieza en la cinta
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
If Range("A2").Value = 1 Then
End If
Nueva_Pieza.Visible = False
Range("A2").Value = 0
Worksheets("OMRON-NJ").TextBox1.Value = "Pulsa MARCHA per començar..."
If Cells(3, 1) = 1 And A_M.Value = True Then
    GoTo Continua
End If
End Sub
Public Sub Reset()
    Range("z15:z25").Interior.Color = RGB(191, 194, 194) 'Color de la cinta
    Range("F15:F25").Interior.Color = RGB(191, 194, 194)
    Range("T15").Interior.Color = xlNone
    Range("P8").Interior.Color = xlNone
    Range("K15").Interior.Color = xlNone
    ActiveSheet.Shapes.Range(Array("Oval 13")).Select
    With Selection.ShapeRange.Fill
        .ForeColor.RGB = RGB(255, 255, 255)
    End With
     ActiveSheet.Shapes.Range(Array("Oval 16")).Select
    With Selection.ShapeRange.Fill
        .ForeColor.RGB = RGB(255, 255, 255)
    End With
    If Range("a1").Value = 1 Then
       ActiveSheet.Shapes.Range(Array("Group 10")).Select
       Selection.ShapeRange.IncrementLeft 150
    End If
    Range("Z25").Activate

    ActiveSheet.Shapes.Range(Array("15 Oval")).Select
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
```

```
    ActiveSheet.Shapes.Range(Array("17 Oval")).Select 'Indicador de pieza en la cinta
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
    ActiveSheet.Shapes.Range(Array("18 Oval")).Select  'Indicador de cilindro arriba
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
    ActiveSheet.Shapes.Range(Array("19 Oval")).Select   'Indicador de cilindro abajo
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)

    Worksheets("OMRON-NJ").TextBox1.Value = "Pulsa MARCHA per començar..."

        Range("Z25").Activate
    If A_M.Value = True Then
        Range("A3").Value = 0
        A_M.Caption = "AUTOMATICO"
        A_M.Picture = LoadPicture("C:\Users\remi_\Downloads\iconos\int-AUT.ico")
        BTN_Reset.Visible = False
        Nueva_Pieza.Visible = False
        A_M_Change

    Else
        A_M.Caption = "MANUAL"
        A_M.Picture = LoadPicture("C:\Users\remi_\Downloads\iconos\int-MAN.ico")
        Range("A1:A6").Value = 0
        A_M.Value = False
    End If

End Sub
```

```
Private Sub Nueva_Pieza_Click()
     Range("A2").Value = 1
     Comprueba_Nueva_Pieza
End Sub
```

```
Public Sub Comprueba_Nueva_Pieza()
If Cells(2, 1) = 1 Then
    ubicacion_celda = 25
    Nueva_Pieza.Visible = False
    Range("A2").Value = 0
    For x = 0 To 10
        ActiveSheet.Shapes.Range(Array("17 Oval")).Select
        Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
        DoEvents
        celda = ubicacion_celda - x
        celda2 = celda + 1
        Range("Z" & celda).Interior.Color = 255
        Range("Z" & celda2).Interior.Color = RGB(191, 194, 194)
        comprueba_Paro
        DoEvents
        Application.Wait (Now + TimeValue("0:00:01"))
        Range("A7").Value = 1
    Next

End If
End Sub
```

```
Public Sub A_M_Change()

Range("A1:A2").Value = 0
Range("A4:A6").Value = 0
Worksheets("OMRON-NJ").TextBox2.Value = Cells(8, 1)
If A_M.Value = True Then
    A_M.Caption = "AUTOMATICO"
    A_M.Picture = LoadPicture("C:\Users\remi_\Downloads\iconos\int-AUT.ico")


        Do While A_M.Value = True
        Worksheets("OMRON-NJ").TextBox1.Value = "Realitzant ordres per veu..."
        Marcha_manual.Enabled = False
```

```
            If Cells(3, 1) = 1 And A_M.Value = True Then
                DoEvents
                Marcha
                Application.Wait (Now + TimeValue("0:00:01"))
            Else
                DoEvents
                Worksheets("OMRON-NJ").TextBox1.Value = "Pulsa MARCHA per començar..."

            End If
        Loop
Else
    A_M.Caption = "MANUAL"
    A_M.Picture = LoadPicture("C:\Users\remi_\Downloads\iconos\int-MAN.ico")
    Marcha_manual.Enabled = True
End If
End Sub
```

```
Public Sub mueve_dos_piezas()
sigue_pieza:
        DoEvents
        Range("A1").Value = 1
        Range("A2").Value = 0
    Range("A7").Value = 0
    ActiveSheet.Shapes.Range(Array("Oval 13")).Select 'Indicador de vectosa en funcionamiento
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)


    Application.Wait (Now + TimeValue("0:00:01"))
    Range("T15").Activate
    ActiveSheet.Shapes.Range(Array("Oval 13")).Select 'Indicador de vectosa en funcionamiento
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)

    Range("Z15").Interior.Color = RGB(191, 194, 194)
    Range("K15").Interior.Color = xlNone
    ActiveSheet.Shapes.Range(Array("Group 10")).Select 'Movimiento del cilindro A
    Selection.ShapeRange.IncrementLeft -150

    Range("T15").Interior.Color = 255
    Range("F15").Interior.Color = 255
    DoEvents
    comprueba_Paro
    'movemos las pieza a la posición original
    Range("a1").Value = 0
    DoEvents
    Application.Wait (Now + TimeValue("0:00:01"))

    ActiveSheet.Shapes.Range(Array("Group 10")).Select 'Movimiento del cilindro A para volver a su posición original
    Selection.ShapeRange.IncrementLeft 150


    Application.Wait (Now + TimeValue("0:00:01"))
    ActiveSheet.Shapes.Range(Array("18 Oval")).Select   'Indicador de cilindro abajo
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)

    Range("T15").Interior.Pattern = xlNone

    ActiveSheet.Shapes.Range(Array("Oval 16")).Select
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)

    Application.Wait (Now + TimeValue("0:00:01"))
    ActiveSheet.Shapes.Range(Array("18 Oval")).Select   'Indicador de cilindro abajo
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
    Range("P8").Interior.Color = 255
    DoEvents
    comprueba_Paro
    Application.Wait (Now + TimeValue("0:00:02"))
    ActiveSheet.Shapes.Range(Array("19 Oval")).Select   'Indicador de cilindro abajo
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
    ActiveSheet.Shapes.Range(Array("Oval 16")).Select
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
    ActiveSheet.Shapes.Range(Array("19 Oval")).Select   'Indicador de cilindro abajo
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
    Range("F14").Interior.Color = RGB(255, 255, 255)
    ubicacion_fcelda = 16
    ActiveSheet.Shapes.Range(Array("17 Oval")).Select
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
    For x = 0 To 9
```

```
        DoEvents
        celdaf = ubicacion_fcelda + x
        celdaf2 = celdaf - 1
        Range("F" & celdaf).Interior.Color = 255
        Range("F" & celdaf2).Interior.Color = RGB(191, 194, 194)
        Application.Wait (Now + TimeValue("0:00:01"))
        comprueba_Paro
    Next
    Range("A8").Value = Range("A8").Value + 1
    Worksheets("OMRON-NJ").TextBox2.Value = Range("A8").Value
    DoEvents

    Range("F25").Interior.Color = RGB(191, 194, 194)
    ActiveSheet.Shapes.Range(Array("17 Oval")).Select
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
    Range("F26").Interior.Color = RGB(255, 255, 255)
    Range("P8").Interior.Pattern = xlNone
    ActiveSheet.Shapes.Range(Array("Oval 13")).Select
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)


    Application.Wait (Now + TimeValue("0:00:01"))
    Range("K15").Activate
    Range("K15").Interior.Color = 255
    DoEvents
    comprueba_Paro
    ActiveSheet.Shapes.Range(Array("Group 10")).Select
    Selection.ShapeRange.IncrementLeft -150
    Range("a1").Value = 1
    DoEvents
    Application.Wait (Now + TimeValue("0:00:01"))
    Range("F15").Activate
    Range("F15").Interior.Color = 255
        DoEvents
    Range("K15").Interior.Pattern = xlNone

    ActiveSheet.Shapes.Range(Array("Oval 13")).Select
    Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
    Application.Wait (Now + TimeValue("0:00:01"))

ubicacion_fcelda = 16
 ActiveSheet.Shapes.Range(Array("17 Oval")).Select
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 0)
For x = 0 To 9
    DoEvents
    celdaf = ubicacion_fcelda + x
    celdaf2 = celdaf - 1
    Range("F" & celdaf).Interior.Color = 255
    Range("F" & celdaf2).Interior.Color = RGB(191, 194, 194)
    Application.Wait (Now + TimeValue("0:00:01"))
    If x = 3 Then
        ActiveSheet.Shapes.Range(Array("Group 10")).Select
        Selection.ShapeRange.IncrementLeft 150
        Range("a1").Value = 0
    End If
    comprueba_Paro
Next
Range("A8").Value = Range("A8").Value + 1
Worksheets("OMRON-NJ").TextBox2.Value = Range("A8").Value
DoEvents

Range("F25").Interior.Color = RGB(191, 194, 194)
Range("F26").Interior.Color = RGB(255, 255, 255)
Range("F27").Interior.Color = xlNone
ActiveSheet.Shapes.Range(Array("17 Oval")).Select
Selection.ShapeRange.Fill.ForeColor.RGB = RGB(255, 255, 255)
Worksheets("OMRON-NJ").TextBox1.Value = "Pulsa MARCHA per començar..."

If Cells(3, 1) = 1 And A_M.Value = True Then
  Marcha
End If

End Sub
```

```
Public Sub comprueba_Paro()

Do While Cells(5, 1) = 1
    BTN_Reset.Visible = True
    DoEvents
    Worksheets("OMRON-NJ").TextBox1.Value = "Esperant ordre de Reset..."
    For i = 1 To 100000
        DoEvents
        If Cells(6, 1) = 1 Then
            Reset
            BTN_Reset.Visible = False
            Nueva_Pieza.Visible = False
            End
        End If
    Next
Loop

End Sub
```